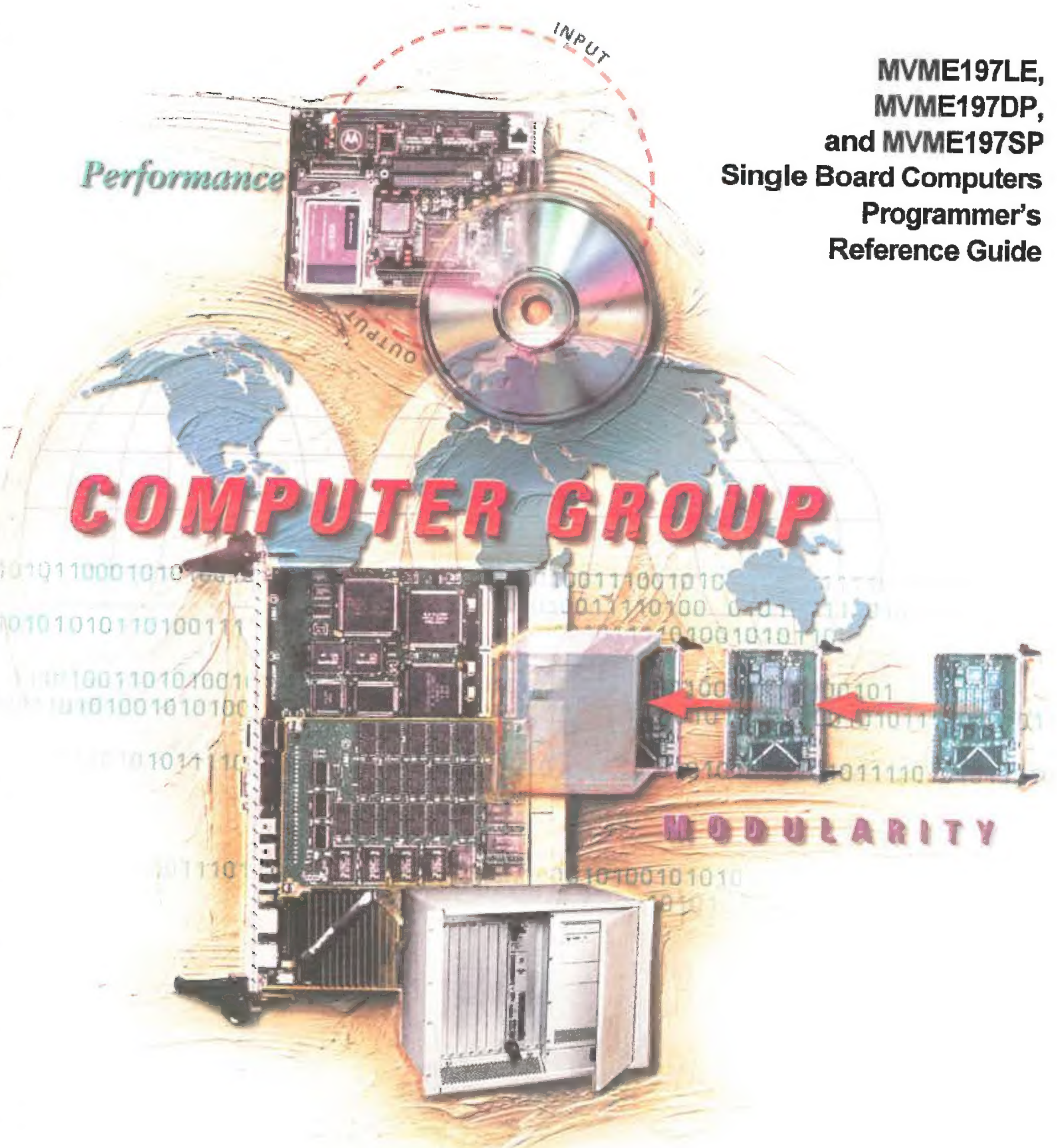
MOTOROLA

MVME197LE, MVME197DP, and MVME197SP Single Board Computers Programmer's Reference Guide



MVME197LE, MVME197DP, and MVME197SP

Single Board Computers

Programmer's Reference Guide

(MVME197PG/D2)

Notice

While reasonable efforts have been made to assure the accuracy of this document, Motorola, Inc. assumes no liability resulting from any omissions in this document, or from the use of the information obtained therein. Motorola reserves the right to revise this document and to make changes from time to time in the content hereof without obligation of Motorola to notify any person of such revision or changes.

It is possible that this publication may contain reference to, or information about Motorola products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that Motorola intends to announce such Motorola products, programming, or services in your country.

Restricted Rights Legend

If the documentation contained herein is supplied, directly or indirectly, to the U.S. Government, the following notice shall apply unless otherwise agreed to in writing by Motorola, Inc.

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

Motorola, Inc.
Computer Group
2900 South Diablo Way
Tempe, Arizona 85282-9602

Preface

This manual will provide board level information, and detailed ASIC chip information including register bit descriptions for the MVME197 series of Single Board Computers.

This manual is intended for anyone who wants to program these boards in order to design OEM systems, supply additional capability to an existing compatible system, or work in a lab environment for experimental purposes.

A basic knowledge of computers and digital logic is assumed.

To use this manual, you should be familiar with the publications listed in the *Related Documentation* section found in the following pages.

Manual Terminology

Throughout this manual, a convention has been maintained whereby data and address parameters are preceded by a character which specifies the numeric format, as follows:

$	dollar	specifies a hexadecimal number
%	percent	specifies a binary number
&	ampersand	specifies a decimal number

For example, "12" is the decimal number twelve, and "$12" is the decimal number eighteen. Unless otherwise specified, all address references are in hexadecimal throughout this manual.

An asterisk (*) following the signal name for signals which are level significant denotes that the signal is true or valid when the signal is low.

An asterisk (*) following the signal name for signals which are edge significant denotes that the actions initiated by that signal occur on high to low transition.

In this manual, *assertion* and *negation* are used to specify forcing a signal to a particular state. In particular, *assertion* and *assert* refer to a signal that is active or true; *negation* and *negate* indicate a signal that is inactive or false. These terms are used independently of the voltage level (high or low) that they represent.

Data and address sizes are defined as follows:

- ❑ A *byte* is eight bits, numbered 0 through 7, with bit 0 being the least significant.
- ❑ A two-byte is 16 bits, numbered 0 through 15, with bit 0 being the least significant. For the MVME197 and other RISC modules, this is called a *half-word*.
- ❑ A four-byte is 32 bits, numbered 0 through 31, with bit 0 being the least significant. For the MVME197 and other RISC modules, this is called a *word*.
- ❑ An eight-byte is 64 bits, numbered 0 through 63, with bit 0 being the least significant. For the MVME197 and other RISC modules, this is called a *double-word*.

Throughout this manual, it is assumed that the MPU on the MVME197 module series is always programmed with *big-endian byte ordering*, as shown below. Any attempt to use *small-endian byte ordering* will immediately render the MVME197Bug debugger unusable.

BIT			BIT
63 56	55 48	47 40	39 32
ADRO	ADR1	ADR2	ADR3
31 24	23 16	15 08	07 00
ADR4	ADR5	ADR6	ADR7

The terms control bit and status bit are used extensively in this document. The term control bit is used to describe a bit in a register that can be set and cleared under software control. The term true is used to indicate that a bit is in the state that enables the function it controls. The term false is used to indicate that the bit is in the state that disables the function it controls. In all tables, the terms 0 and 1 are used to describe the actual value that should be written to the bit, or the value that it yields when read. The term status bit is used to describe a bit in a register that reflects a specific condition. The status bit can be read by software to determine operational or exception conditions.

Related Documentation

The following publications are applicable to the MVME197 module series and may provide additional helpful information. If not shipped with this product, they may be purchased by contacting your Motorola sales office.

Document Title	Motorola Publication Number
MVME197LE Single Board Computer User's Manual	MVME197LE
MVME197LE Single Board Computer Support Information	SIMVME197LE
MVME197DP and MVME197SP Single Board Computer User's Manual	MVME197
MVME197DP and MVME197SP Single Board Computer Support Information	SIMVME197
MVME197BUG 197Bug Debugging Package User's Manual	MVME197BUG
MVME197BUG 197Bug Diagnostic Firmware User's Manual	MVME197DIAG
MVME712M Transition Module and P2 Adapter Board User's Manual	MVME712M
MVME712-12, MVME712-13, MVME712A, MVME712AM, and MVME712B Transition Module and LCP2 Adapter Board User's Manual	MVME712A
MC88110 Second Generation RISC Microprocessor User's Manual	MC88110UM
MC68040 Microprocessor User's Manual	MC68040UM
MC88410 Secondary Cache Controller User's Manual	MC88410UM

Note

Although not shown in the above list, each Motorola Computer Group manual publication number is suffixed with characters which represent the revision level of the document, such as "/D2" (the second revision of a manual); a supplement bears the same number as the manual but has a suffix such as "/A1" (the first supplement to the manual).

To further assist your development effort, Motorola has collected user's manuals for each of the peripheral controllers used on the MVME197 module series and other boards from the suppliers. This bundle includes manuals for the following:

68-1X7DS for use with the MVME197 series of Single Board Computers.

NCR 53C710 SCSI Controller Data Manual and Programmer's Guide
Intel i82596 Ethernet Controller User's Manual
Cirrus Logic CD2401 Serial Controller User's Manual
SGS-Thompson MK48T08 NVRAM/TOD Clock Data Sheet

The following non-Motorola publications may also be of interest and may be obtained from the sources indicated. The VMEbus Specification is contained in ANSI/IEEE Standard 1014-1987.

ANSI/IEEE Std 1014-1987
Versatile Backplane Bus: VMEbus

The Institute of Electrical and Electronics
Engineers, Incorporated
Publication and Sales Department
345 East 47th Street
New York, New York 10017-2633
Telephone: 1-800-678-4333

ANSI Small Computer System Interface-2
(SCSI-2), Draft Document X3.131-198X,
Revision 10c

Global Engineering Documents
P.O. Box 19539
Irvine, California 92713-9539
Telephone (714) 979-8135

Printed in the United States of America

January 1994

Contents

CHAPTER 1 BOARD LEVEL HARDWARE DESCRIPTION

CHAPTER 2 VMEchip2

CHAPTER 4 BUSSWITCH

CHAPTER 5 DRAM CONTROLLER AND ADDRESS MULTIPLEXER (DCAM)

CHAPTER 6 ERROR CORRECTION AND DATA MULTIPLEXER (ECDM)

CHAPTER 7 PRINTER AND SERIAL PORT CONNECTIONS

List of Figures

List of Tables

BOARD LEVEL HARDWARE DESCRIPTION 1

Introduction

This manual provides programming information for the MVME197 series of Single Board Computers. Extensive programming information is provided for the Application-Specific Integrated Circuit (ASIC) devices used on the MVME197 module series. Refer to Table 1-1 for model designations.

This chapter describes the board level hardware features of the MVME197 series of single board computers, hereafter referred to as the MVME197, unless separately specified. The chapter is organized with a board level overview and features listed in this introduction, followed by a more detailed hardware functional description. Front panel switches and indicators are included in the detailed hardware functional description. Memory maps are next, and the chapter closes with some general software considerations.

All programmable registers in the MVME197 reside in ASIC devices and are covered in device specific chapters. Chapter 2 covers the VMEchip2 (VME2), Chapter 3 covers the PCCchip2 (PCC2), Chapter 4 covers the BusSwitch gate array, Chapter 5 covers the DCAM (DRAM Controller and Address Multiplexer), and Chapter 6 covers the ECDM (Error Correction and Data Multiplexer). For those interested in programmable register bit definitions and less interested in hardware functionality, focus on Chapters 2, 3, 4, 5, and 6. In some cases, however, Chapter 1 gives related background information.

Model Designations

The MVME197 Single Board Computer is avai lable in the following models.

Table 1-1. MVME197 Model Designations

Model Number	Motorola Part Number	Model Description
MVME197LE series		
MVME197-001	01-W3869B03L	Single Processor MC88110, 32MB Onboard ECC DRAM, 50 MHz
MVME197-002	01-W3869B04L	Single Processor MC88110, 64MB Onboard ECC DRAM, 50 MHz

Table 1-1. MVME197 Model Designations (Continued)

Model Number	Motorola Part Number	Model Description
MVME197LE series		
MVME197-003	01-W3869B01M	Single Processor MC88110, 32MB Onboard ECC DRAM, 40 MHz
MVME197-004	01-W3869B02M	Single Processor MC88110, 64MB Onboard ECC DRAM, 40 MHz
MVME197SP series		
MVME197-101	01-W3815B02	Single Processor MC88110, 256KB Cache, 128MB Onboard ECC DRAM, 50 MHz
MVME197DP series		
MVME197-201	01-W3815B01	Dual Processor MC88110, 256KB Cache per Processor, 128MB Onboard ECC DRAM, 50 MHz
MVME197-202	01-W3815B03	Dual Processor MC88110, 256KB Cache per Processor, 256MB Onboard ECC DRAM, 50 MHz

Overview

Each MVME197 module is a double-high VMEmodule based on the MC88110 RISC microprocessor.

The **MVME197LE** series have 32/64MB of DRAM, 1MB of flash memory, 8KB of static RAM (with battery backup), a time of day clock (with battery backup), an Ethernet transceiver interface, four serial ports with EIA-232-D interface, six tick timers, a watchdog timer, 128/256KB of BOOT ROM, a SCSI bus interface with DMA (Direct Memory Access), a Centronics printer port, an A16/A24/A32/D8/D16/D32 VMEbus master/slave interface, and a VMEbus system controller.

The **MVME197DP/SP** series have 128/256MB of DRAM, 256KB of cache memory for **each** MC88110/MC88410 microprocessor/cache controller combination (note that the MVME197SP version has only one MC88110/MC88410 device combination), 1MB of flash memory, 8KB of static RAM (with battery backup), a time of day clock (with battery backup), an Ethernet transceiver interface, four serial ports with EIA-232-D interface, six tick timers, a watchdog timer, 128/256KB of BOOT ROM, a SCSI bus interface with DMA (Direct Memory Access), a Centronics printer port, an A16/A24/A32/D8/D16/D32 VMEbus master/slave interface, and a VMEbus system controller.

Input/Output (I/O) signals are routed through the MVME197's backplane connector P2. A P2 Adapter Board or LCP2 Adapter board routes the signals and grounds from connector P2 to an MVME712 series transition module. The MVME197 supports the MVME712M, MVME712A, MVME712AM, and MVME712B transition boards (referred to here as the MVME712X, unless separately specified). The MVME197 also supports the MVME712-12 and MVME712-13 (referred to as the MVME712-XX, unless separately specified). These transition boards provide configuration headers, serial port drivers, and industry standard connectors for the I/O devices.

All MVME197 modules have eight ASIC devices (described in the following order: BusSwitch, DCAM, ECDM, PCC2, and VME2).

Note **For the MVME197 series, the term Local Bus, as used in other MVME1xx Single Board Computer series, is referred to as the Local Peripheral Bus.**

The BusSwitch ASIC provides an interface between the processor bus (MC88110 bus) and the local peripheral bus (MC68040 compatible bus). Refer to the board specific MVME197 block diagram (Figures 1-1, 1-2, and 1-3). It provides bus arbitration for the MC88110 bus and serves as a seven level interrupt handler. It has programmable map decoders for both busses, as well as write post buffers on each, two tick timers, and four 32-bit general purpose registers.

The DCAM (DRAM Controller and Address Multiplexer) ASIC provides the address multiplexers and RAS/CAS/WRITE control for the DRAM as well as data control for the ECDM.

The ECDM (Error Correction and Data Multiplexer) ASIC multiplexes between four data paths on the DRAM array. Since the device handles 16 bits, four such devices are required on the MVME197 to accommodate the 64-bit data bus of the MC88110 microprocessor. Single-bit error correction and double-bit detection is performed in the ECDM.

The PCCchip2 (Peripheral Channel Controller) ASIC provides two tick timers and the interface to the LAN chip, the SCSI chip, the serial port chip, the printer port, and the BBRAM (Battery Backup RAM).

The VMEchip2 ASIC provides a VMEbus interface. The VMEchip2 includes two tick timers, a watchdog timer, programmable map decoders for the master and slave interfaces, and a VMEbus to/from the local peripheral bus DMA controller, a VMEbus to/from the local peripheral bus non-DMA programmed access interface, a VMEbus interrupter, a VMEbus system controller, a VMEbus interrupt handler, and a VMEbus requester.

Local peripheral bus to VMEbus transfers can be D8, D16, or D32. VMEchip2 DMA transfers to the VMEbus, however, can be 64 bits wide as Block Transfer (BLT).

Requirements

These boards are designed to conform to the requirements of the following documents:

- ❑ VMEbus Specification (IEEE 1014-87)
- ❑ EIA-232-D Serial Interface Specification, EIA
- ❑ SCSI Specification, ANSI

Features

These are some of the major features of the MVME197 module series:

- ❑ Single MC88110 RISC Microprocessor (MVME197LE modules)
- ❑ Single MC88110 RISC Microprocessor with an MC88410 Cache Controller (MVME197SP modules)
- ❑ Dual MC88110 RISC Microprocessors, each processor with one MC88410 Cache Controller (MVME197DP modules)
- ❑ 256 kilobytes of external cache per processor, controlled by the MC88410 Cache Controller (MVME197DP and MVME197SP modules)
- ❑ 32 or 64 megabytes of 64-bit Dynamic Random Access Memory (DRAM) with Error Checking and Correction (ECC) (MVME197LE modules)
- ❑ 128 or 256 megabytes of 64-bit Dynamic Random Access Memory (DRAM) with Error Checking and Correction (ECC) (MVME197DP and MVME197SP modules)
- ❑ 1 megabyte of Flash memory
- ❑ Six status LEDs (FAIL, RUN, SCON, LAN, SCSI, and VME)
- ❑ 8 kilobytes of Static Random Access Memory (SRAM) and Time of Day (TOD) clock with battery backup RAM (BBRAM)
- ❑ Two push-button switches (ABORT and RESET)
- ❑ 128 or 256 kilobytes of BOOT ROM
- ❑ Six 32-bit tick timers for periodic interrupts
- ❑ Watchdog timer
- ❑ Eight software interrupts

- ❑ I/O
  - SCSI Bus interface with Direct Memory Access (DMA)
  - Four serial ports with EIA-232-D buffers
  - Centronics printer port
  - Ethernet transceiver interface
- ❑ VMEbus interface
  - VMEbus system controller functions
  - VMEbus interface to local peripheral bus (A24/A32, D8/D16/D32 BLT (D8/D16/D32/D64))(BLT = Block Transfer)
  - Local peripheral bus to VMEbus interface (A24/A32, D8/D16/D32 BLT (D16/D32/D64))
  - VMEbus interrupter
  - VMEbus interrupt handler
  - Global CSR for inter-processor communications
  - DMA for fast local memory - VMEbus transfers (A16/A24/A32, D16/D32 BLT (D16/D32/D64))

Block Diagrams

General block diagrams for the MVME197 series of single board computers are provided as follows. Figure 1-1 illustrates the block diagram for the MVME197LE, Figure 1-2 illustrates the block diagram for the MVME197SP, and Figure 1-3 illustrates the block diagram for the MVME197DP.

Functional Description

This section contains a functional description of the major blocks on the MVME197 series of single board computers.

Front Panel Switches and Indicators

There are two push-button switches and six LEDs (Light Emitting Diodes) on the front panel of the MVME197. The switches are RESET and ABORT. The RESET switch (S3) will reset all onboard devices and drive the SYSRESET* signal if the board **is** the system controller. The RESET switch (S3) will reset all onboard devices except the DCAM and ECDM if the board **is not** the system controller. The VMEchip2 generates the SYSRESET* signal. The BusSwitch combines the VMEchip2 local reset, the power up reset, and the reset switch to generate a local board reset. Refer to the *Reset Module* section in the *BusSwitch* chapter for more information.

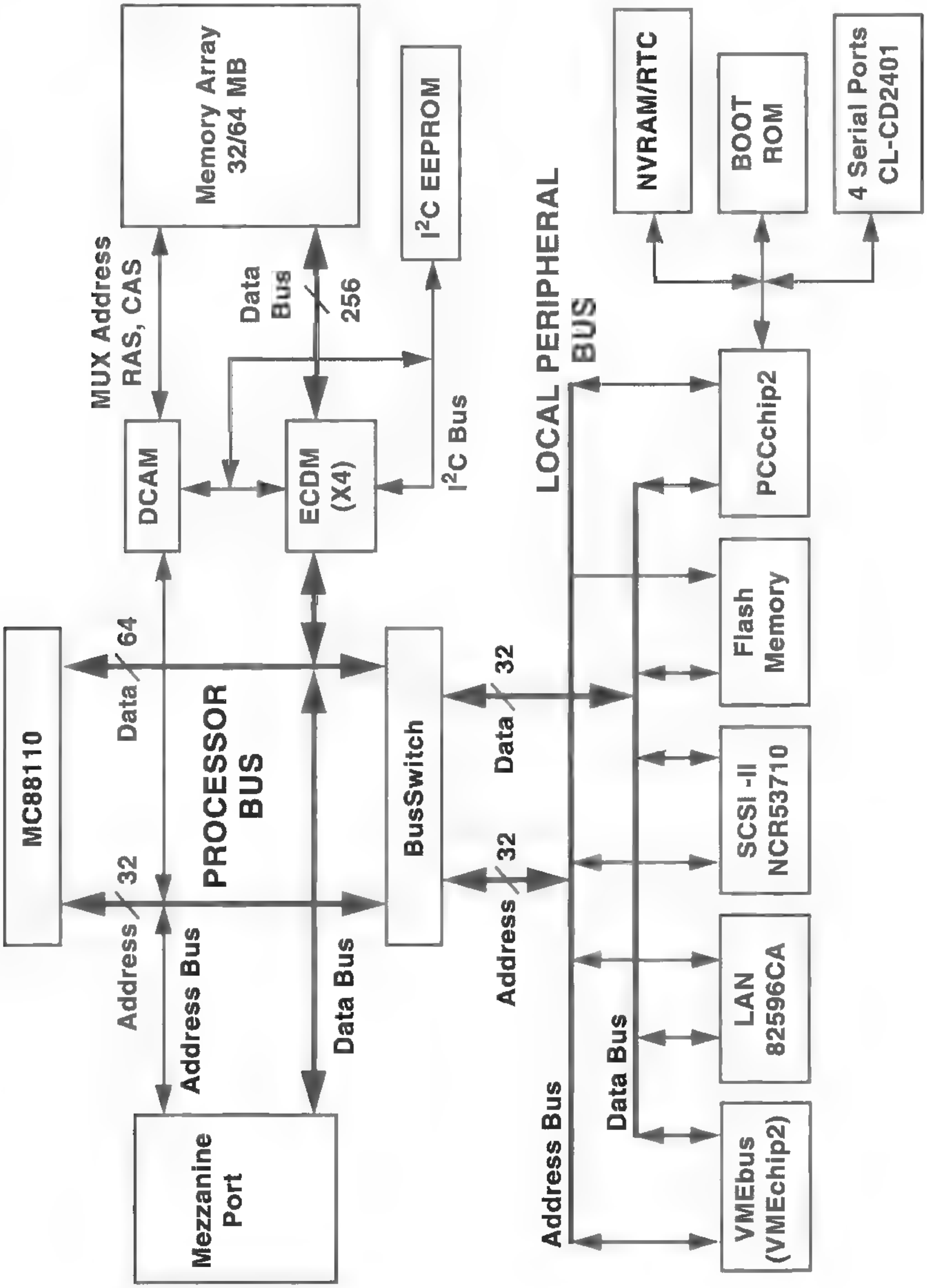


Figure 1-1. MVME197LE Block Diagram

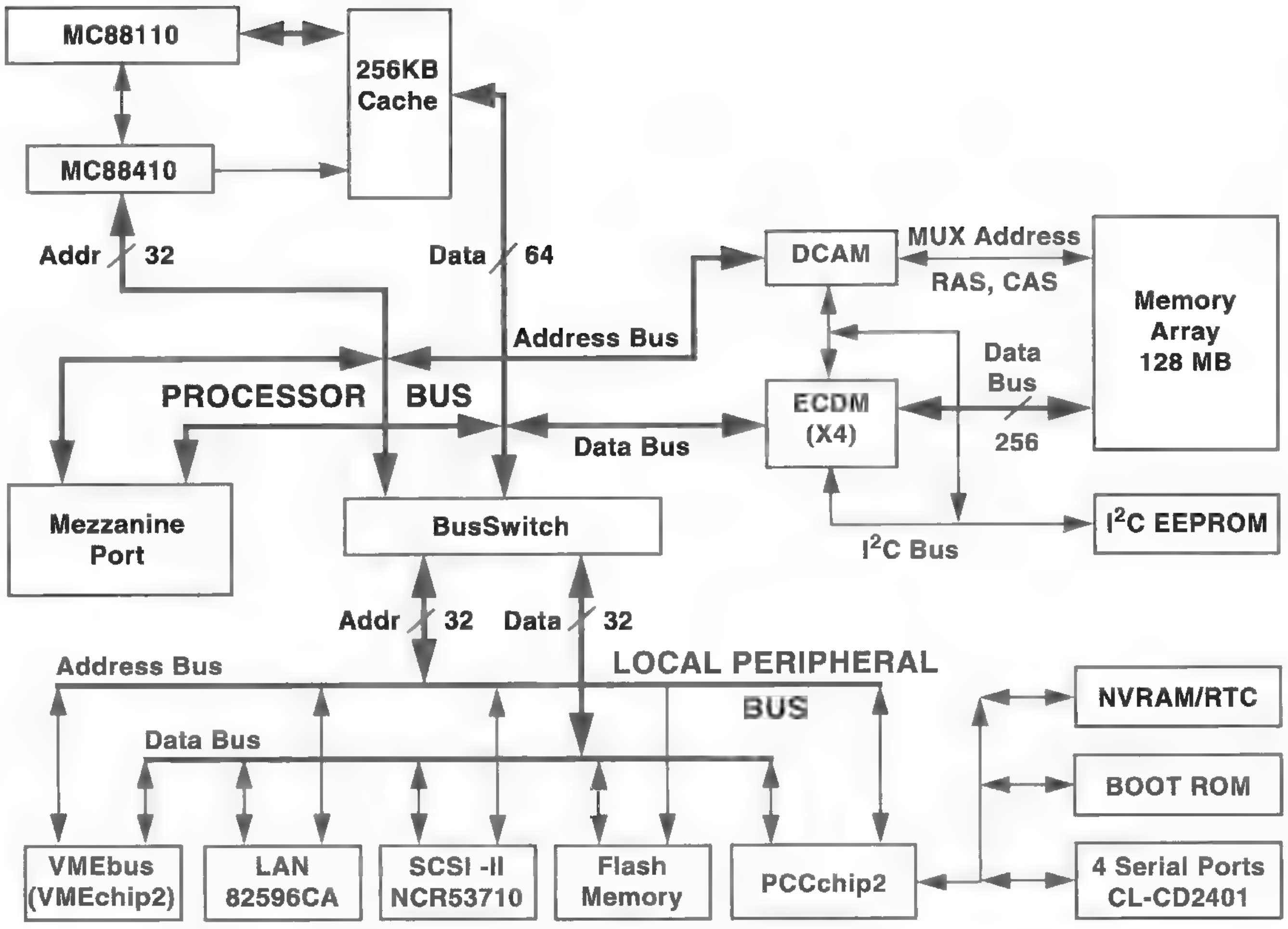


Figure 1-2. MVME197SP Block Diagram

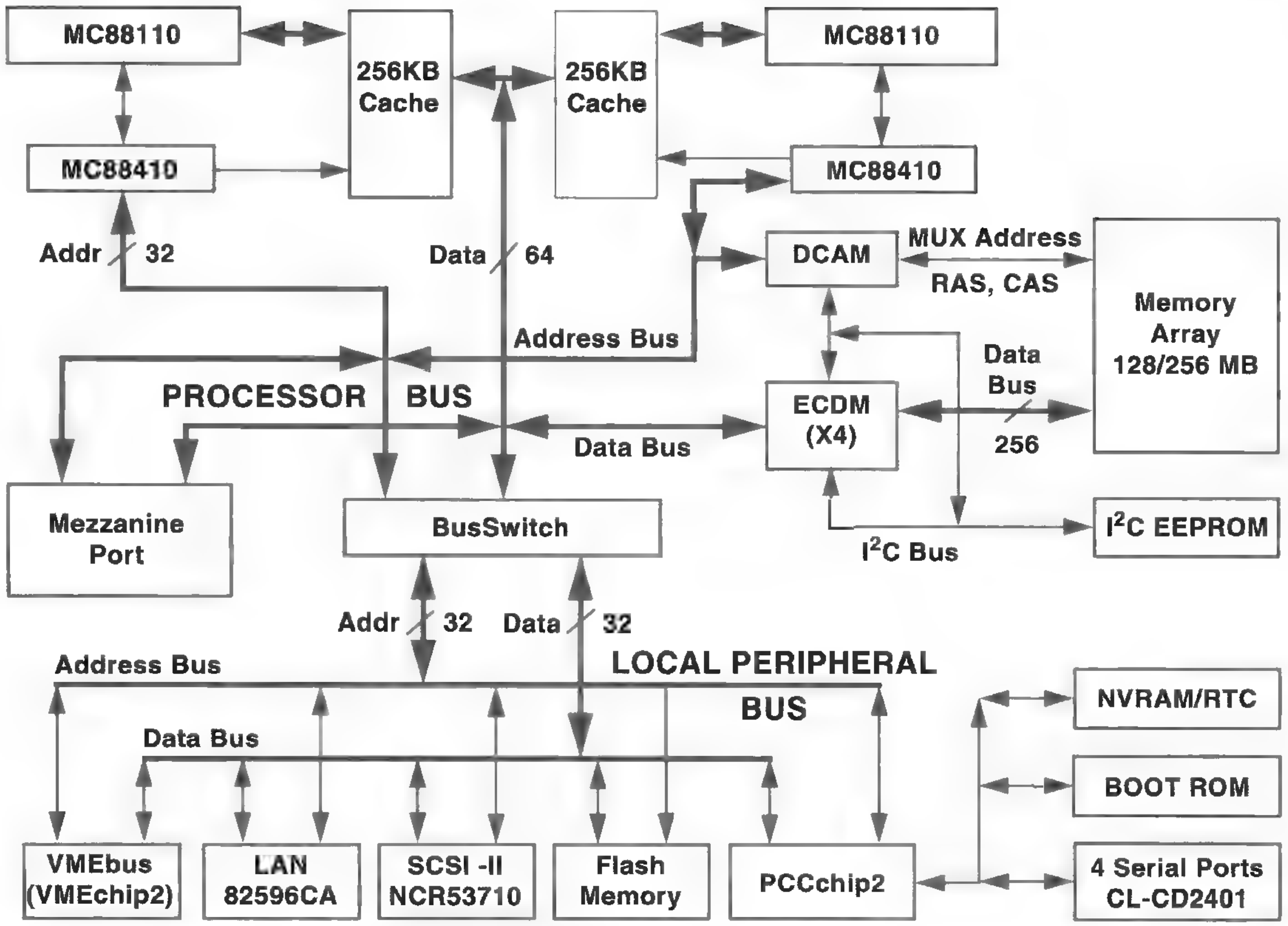


Figure 1-3. MVME197DP Block Diagram

The ABORT switch (S2) can generate an interrupt to CPU0 via the NMI* signal. It is normally used to abort program execution and return to the debugger. This capability is controlled via the ABORT register in the BusSwitch.

The six LEDs on the MVME197 front panel are: FAIL, SCON, RUN, LAN, VME, and SCSI.

The yellow FAIL LED (DS1) is lit when the BRDFAIL signal line is active.

The green SCON LED (DS2) is lit when the VMEchip2 is the VMEbus system controller.

The green RUN LED (DS3) is lit when the MC88110 bus MC* pin is low.

The green LAN LED (DS4) lights when the LAN chip is the local peripheral bus master.

The green VME LED (DS5) lights when the board is using the VMEbus or when the board is accessed by the VMEbus.

The green SCSI LED (DS6) lights when the SCSI chip is the local peripheral bus master.

Data Bus Structure

The data bus structure is arranged to accommodate the various 8-bit, 16-bit, 32-bit, and 64-bit devices that reside on the module. Refer to the specific section of this manual and to the user's guide for each device to determine its port size, data bus connection, and any restrictions that apply when accessing the device.

MC88110 MPU

The MVME197 series of single board computers are based on the MC88000 families of RISC (Reduced Instruction Set Computer) microprocessors. Depending on the specific MVME197 module, the MVME197 series uses the MC88110 RISC microprocessor. Refer to the *Module Designation* section in the beginning of this chapter for MVME197 module/processor variations and to the *MC88110 Second Generation RISC Microprocessor User's Manual* for more detailed information on this device.

MC88410 Cache Controller

(The following text is applicable only to the MVME197DP/SP versions of the MVME197 series).

Depending on the specific MVME197DP/SP module version, each MC88110 microprocessor is connected directly to an MC88410 Secondary Cache

Controller. Each MC88410 controls a 256KB level two cache. Refer to the *MC88410 Secondary Cache Controller User's Manual* and the *MCM62110 Data Sheet* for more information on this device.

BOOT ROM

The board accommodates a 32-pin PLCC/CLCC ROM/EPROM referred to as BOOT ROM or DROM (Download ROM). It is organized as a 128/256K x 8 device, but as viewed from the processor it looks like a 16/32K x 64 memory. This memory is mapped starting at location $FFF80000, but after a local reset it is also mapped at location 0, providing a reset vector and bootstrap code for the processor. The DR0 bit in the General Control Register (GCR) of the PCCchip2 must be cleared to disable the BOOT ROM memory map at 0. In addition, the ROM0 bit in the ROMCR register of the BusSwitch must be cleared.

Flash Memory

Up to 1MB of flash memory is available on the board. Flash memory works like EPROM, but can be erased and reprogrammed by software. It is organized as 32 bits wide, but to the processor it looks as 64 bits wide. It is mapped at location $FF800000. Reads can be of any size, including burst transfers, but writes are always 32 bits wide, regardless of the size specified for the transfer. For this reason, software should only use 32-bit write transfers. This memory is controlled by the BusSwitch, and the memory size, access time, and write enable capability can be programmed via the ROM Control Register (ROMCR) in the BusSwitch. The flash memory can be accessed from the processor bus only. It is not accessible from the local peripheral bus or VMEbus.

Onboard DRAM

The onboard DRAM on the MVME197LE (2 banks of 32MB memory, one optionally installed) is sized at 32MB using 1M x 4 devices and configured as 256 bits wide.

The onboard DRAM on the MVME197DP/SP (2 banks of 128MB memory, one optionally installed) is sized at 128MB using 4M x 4 devices and configured as 256 bits wide.

The DRAM is four-way interleaved to efficiently support cache burst cycles. The DRAM is controlled by the DCAM and ECDM, and the map decoders in the DCAM can be programmed through the I^2C bus interface in the ECDM to accommodate different base address(es) and sizes. The onboard DRAM is not disabled by a local peripheral bus reset. Refer to the *DCAM* and *ECDM* chapters for detailed programming information.

Battery Backup RAM and Clock

The MK48T08 RAM and clock chip is used on the MVME197. This chip provides a time of day clock, oscillator, crystal, power fail detection, memory write protection, 8KB of RAM, and a battery in one 28-pin package. The clock provides seconds, minutes, hours, day, date, month, and year in BCD 24-hour format. Corrections for 28-, 29-, (leap year) and 30-day months are automatically made. No interrupts are generated by the clock. The MK48T08 is an 8-bit device; however the interface provided by the PCCchip2 supports 8-, 16-, and 32-bit accesses to the MK48T08. Refer to the *PCCchip2* chapter and to the MK48T08 data sheet for detailed programming information.

VMEbus Interface

The local peripheral bus to VMEbus interface, the VMEbus to local peripheral bus interface, and the local-VMEbus DMA controller functions on the MVME197 are provided by the VMEchip2. The VMEchip2 can also provide the VMEbus system controller functions. Refer to the *VMEchip2* chapter for detailed programming information.

I/O Interfaces

The MVME197 provides onboard I/O for many system applications. The I/O functions include serial ports, a printer port, an Ethernet transceiver interface, and a SCSI mass storage interface.

Serial Port Interface

The CD2401 serial controller chip (SCC) is used to implement the four serial ports. The serial ports support the standard baud rates (110 to 38.4K baud). Serial port 4 also supports synchronous modes of operation.

All four of the serial ports are different functionally because of the limited number of pins on the I/O connector. Serial port 1 is a minimum function asynchronous port. It uses RXD, CTS, TXD, and RTS. Serial ports 2 and 3 are full function asynchronous ports. They use RXD, CTS, DCD, TXD, RTS, and DTR. Serial port 4 is a full function asynchronous or synchronous port. It can operate at synchronous bit rates up to 64k bits per second. It uses RXD, CTS, DCD, RTS, and DTR. It also interfaces to the synchronous clock signal lines. Refer to *Printer and Serial Port Connections* chapter for drawings of the serial port interface connections.

All four serial ports use EIA-232-D drivers and receivers located on the main board, and all the signal lines are routed to the I/O connector. The configuration headers are located on the MVME712X transition board. An external I/O transition board such as the MVME712X should be used to convert the I/O connector pinout to industry-standard connectors.

The interface provided by the PCCchip2 allows the 16-bit CD2401 to appear at contiguous addresses; however, accesses to the CD2401 must be 8 or 16 bits. 32-bit accesses are not permitted. Refer to the CD2401 data sheet and to the *PCCchip2* chapter for detailed programming information.

The CD2401 supports DMA operations to local memory. Because the CD2401 does not support a retry operation necessary to break VMEbus lock conditions, the CD2401 DMA controllers should not be programmed to access the VMEbus. The hardware does not restrict the CD2401 to onboard DRAM.

Printer Interface

The MVME197 has a Centronics-compatible printer interface. The printer interface is provided by the PCCchip2. Refer to the *PCCchip2* chapter for detailed programming information. Refer to *Printer and Serial Port Connections* chapter for drawings of the serial port interface connections.

Ethernet Interface

The 82596CA is used to implement the Ethernet transceiver interface. The 82596CA accesses local RAM using DMA operations to perform its normal functions. Because the 82596CA has small internal buffers and the VMEbus has an undefined latency period, buffer overrun may occur if the DMA is programmed to access the VMEbus. Therefore, the 82596CA should not be programmed to access the VMEbus.

Every MVME197 module is assigned an Ethernet Station Address. This address is $08003E2XXXXX, where XXXXX is the unique 5-nibble number assigned to the board (i.e., every MVME197 has a different value for XXXXX).

The Ethernet Station Address is displayed on a label attached to the VMEbus P2 connector. In addition, the eight bytes including the Ethernet address are stored in the configuration area of the BBRAM, with the two lower bytes of those set to 0. That is, 08003E2XXXXX0000 is stored in the BBRAM. At an address of $FFFC1F2C, the upper four bytes (08003E2X) can be read. At an address of $FFFC1F30, the lower four bytes (XXXX0000) can be read. Refer to the BBRAM, TOD Clock memory map description later in this chapter. The MVME197 debugger has the capability to retrieve or set the Ethernet address.

If the data in the BBRAM is lost, the user should use the number on the VMEbus P2 connector label to restore it. Refer to the *197Bug Debugging Package User's Manual*.

The Ethernet transceiver interface is located on the MVME197 main module, and the industry standard connector is located on the MVME712X transition module.

Support functions for the 82596CA are provided by the PCCchip2. Refer to the *82596CA LAN Coprocessor User's Manual* and to the *PCCchip2* chapter for detailed programming information.

SCSI Interface

The MVME197 provides for mass storage subsystems through the industry-standard SCSI bus. These subsystems may include hard and floppy disk drives, streaming tape drives, and other mass storage devices. The SCSI interface is implemented using the NCR 53C710 SCSI I/O controller.

Support functions for the 53C710 are provided by the PCCchip2. Refer to the *NCR 53C710 SCSI I/O Processor Data Manual* and to the *PCCchip2* chapter for detailed programming information.

SCSI Termination

The system configurer must ensure that the SCSI bus is terminated properly. On the MVME197, the terminators are located on the P2 transition board. The +5V power to the SCSI bus termination resistors is provided by the P2 transition board.

Peripheral Resources

The MVME197 includes many resources for the local processor. These include tick timers, software programmable hardware interrupts, watchdog timer, and local peripheral bus timeout.

Programmable Tick Timers

Six 32-bit programmable tick timers with 1 µsec resolution are provided, two in the BusSwitch, two in the VMEchip2, and two in the PCCchip2. The tick timers can be programmed to generate periodic interrupts to the processor. Refer to the *VMEchip2*, *PCCchip2*, and *BusSwitch* chapters for detailed programming information.

Watchdog Timer

A watchdog timer function is provided in the VMEchip2. When the watchdog timer is enabled, it must be reset by software within the programmed time or it times out. The watchdog can be programmed to generate a SYSRESET* signal, local reset signal, or board fail if it times out. Refer to the *VMEchip2* chapter for detailed programming information.

Software-Programmable Hardware Interrupts

Eight software-programmable hardware interrupts are provided by the VMEchip2. These interrupts allow software to create a hardware interrupt. Refer to the *VMEchip2* chapter for detailed programming information.

Processor Bus Timeout

The BusSwitch provides a bus timeout circuit for the processor bus. When enabled by the BTIMER register in the BusSwitch, the timer starts counting when DBB* is asserted, and if the cycle is not terminated (TA*, TEA*, or TRTRY* asserted) before the programmed timeout period, TEA* is asserted. This timer is disabled if the access goes to the local peripheral bus.

Local Peripheral Bus Timeout

The MVME197 provides a timeout function for the processor bus (MC88110 bus) and for the local peripheral bus (MC68040 compatible bus). When the timer is enabled and a bus access times out, a Transfer Error Acknowledge (TEA) signal is generated. The timeout value is selectable by software for 8 μsec, 64 μsec, 256 μsec, or infinite for the local peripheral bus. The local peripheral bus timer does not operate during VMEbus bound cycles. VMEbus bound cycles are timed by the VMEbus access timer and the VMEbus global timer.

Interrupt Sources

MVME197 MPU interrupts are channeled through the BusSwitch. They may come from internal BusSwitch sources as well as from the PCCchip2 (IPL inputs to the BusSwitch), the VMEchip2 (XIPL inputs to the BusSwitch), and other external sources (PALINT and IRQ). The BusSwitch may also generate the non-maskable interrupt (NMI) signal to the MPU from the ABORT push-button switch. Refer to the *BusSwitch*, *PCCchip2*, and *VMEchip2* chapters for more detailed information.

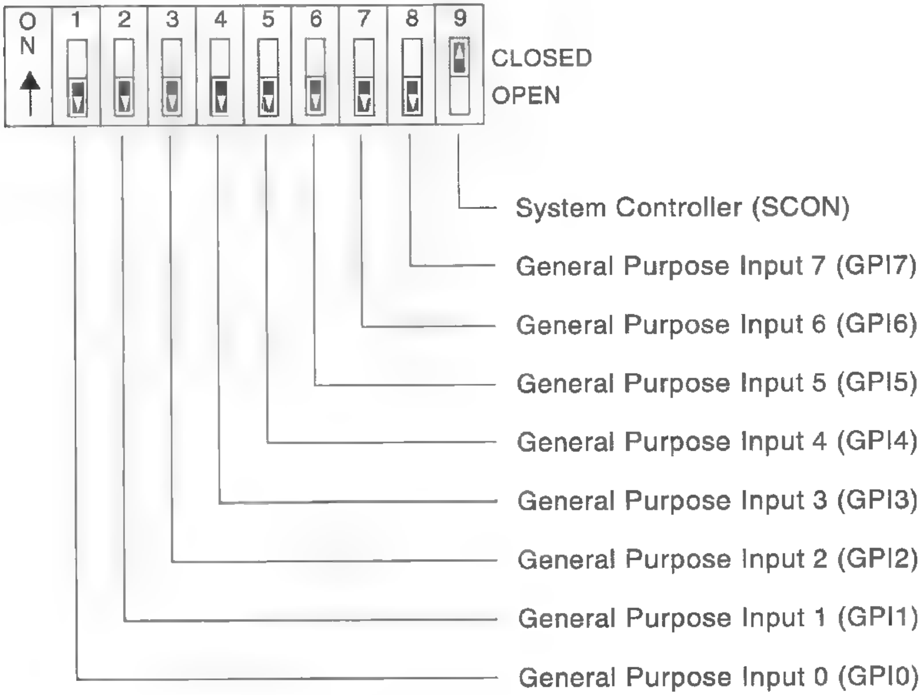
Configuration Switches

The MVME197 was designed to provide software control for most options. However, some options can not be done in software, and are therefore configured by selecting various switch settings.

Configuration Switch S1: General Information

Switch S1 is a bank of nine two-way switch segments. The following illustration shows the factory configuration of switch S1. The bit values are read as a one when the switch is **OFF** (open), and as a zero when the switch is **ON** (closed). The default value for switch S1 is shown below.

Switch S1

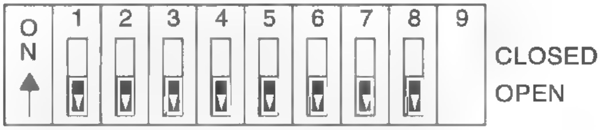

ON
1
2
3
4
5
6
7
8
9
CLOSED
OPEN
System Controller (SCON)
General Purpose Input 7 (GPI7)
General Purpose Input 6 (GPI6)
General Purpose Input 5 (GPI5)
General Purpose Input 4 (GPI4)
General Purpose Input 3 (GPI3)
General Purpose Input 2 (GPI2)
General Purpose Input 1 (GPI1)
General Purpose Input 0 (GPI0)

(FACTORY CONFIGURATION)

Configuration Switch S1: General Purpose Functions (S1-1 to S1-8)

The eight General Purpose Input lines (GPI0-GPI7) on the MVME197 may be configured with selectable switch segments S1-1 through S1-8. These switches can be read as a register (at $FFF40088) in the VMEchip2 LCSR. Refer to the *VMEchip2* chapter for the status of lines GPI0 through GPI7. Factory configuration is with the general purposes input lines disabled (open).

Switch S1

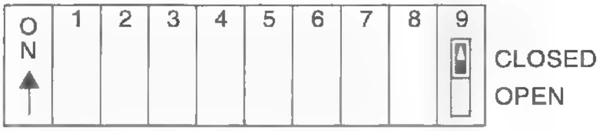


S1-1 to S1-8: **OFF** -- All Ones (FACTORY CONFIGURATION)

Configuration Switch S1: System Controller Enable Function (S1-9)

The MVME197 can be the system controller. The system controller function is enabled or disabled by configuring selectable switch segment S1-9. When the MVME197 is the system controller, the SCON LED is turned **ON**. The VMEchip2 may be configured as a system controller as illustrated below. Factory configuration is with the system controller switch enabled (closed).

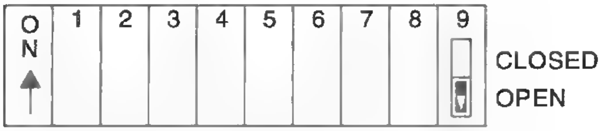
Switch S1



S1-9: **ON** -- MVME197 **<u>IS</u>** the System Controller

(FACTORY CONFIGURATION)

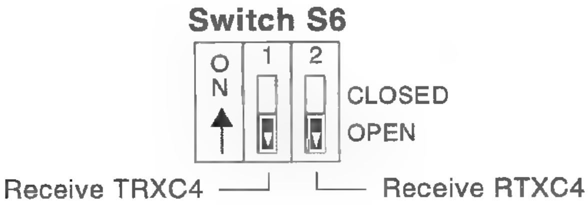
Switch S1



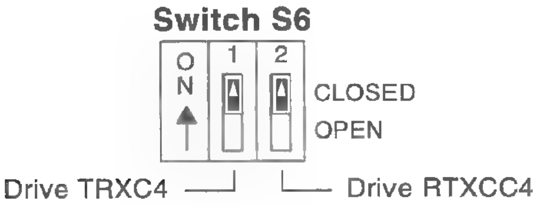
S1-9: **OFF** -- MVME197 **<u>IS NOT</u>** the System Controller

Configuration Switch S6: Serial Port 4 Clock Select (S6-1, S6-2)

Serial port 4 can be configured to use clock signals provided by the RTXC4 and TRXC4 signal lines. Switch segments S6-1 and S6-2 on the MVME197 configures serial port 4 to drive or receive TRXC4 and RTXC4, respectively. Factory configuration is with serial port 4 set to receive both signals (open). The remaining configuration of the clock lines is accomplished by using the Serial Port 4 Clock Configuration Select header on the MVME712M transition module. Refer to the *MVME712M Transition Module and P2 Adapter Board User's Manual* for configuration of that header.



(FACTORY CONFIGURATION)



Connectors

The MVME197 has two 64-position DIN connectors: P1 and P2. Connector P1 rows A, B, C, and connector P2 row B provide the VMEbus interconnection. Connector P2 rows A and C provide the interconnect to the SCSI bus, the serial ports, the Ethernet interface, and the Centronics printer.

The 249-pin mezzanine connector (see note below) provides the MVME197 modules with an MC88110 bus interface for expansion memory. 32MB to 256MB expansion memory mezzanine modules are available for field upgrades. The MVME197 supports the addition of up to two memory expansion modules. Each module requires a VMEbus slot but does not physically connect to the VMEbus. The expansion memory has the same high speed access as the onboard main memory.

Note **The MVME197LE module series and the MVME197DP/SP module series are different artworks. On the MVME197LE series, the mezzanine connector is designated J2, while on the MVME197DP/SP series, the same mezzanine connector is designated J1. The basic form, fit, and function of this mezzanine connector is not changed.**

On the MVME197LE module series there is also a 20-pin general purpose connector (J1) which provides the interconnect to the LEDs and the reset and abort signals. This connector is used only in the MVME197LE series.

On the MVME197DP/SP module series there is a bank of ten two-way switch segments which is designated connector J4. This connector switch is not used and all ten switch segments are soldered over.

Refer to the board specific SIMVME197 support information manual for detailed connector signal descriptions.

Memory Maps

There are four points of view for the memory maps: 1) the mapping of all resources as viewed by each processor bus (PAA/PDA-PAB/PDB bus), 2) the mapping of onboard/offboard resources as viewed from the Local Peripheral Bus (MC68040 compatible), 3) the mapping of all resources as viewed by each of the MC88410 Cache Controllers (PA/PD bus), and 4) the mapping of onboard resources as viewed by VMEbus Masters (VMEbus memory map).

Processor Bus Memory Map

Care should be taken, since memory maps 2, 3, and 4 are programmable. It is recommended that direct mapping from the PA/PD Bus to the Local Peripheral Bus be used.

The memory maps of MVME197 devices are provided in the following tables. Table 1-2 is the entire map from $00000000 to $FFFFFFFF. Many areas of the map are user-programmable, and suggested uses are shown in the table. This is assuming no address translation is used between the PA/PD bus and local peripheral bus and between the local peripheral bus and VMEbus. The cache inhibit function is programmable in the MC88110 microprocessor. The onboard I/O space must be marked cache inhibit and serialized in its page table. Table 1-3 further defines the map for the local devices.

Table 1-2. Processor Bus Memory Map

Address Range	Devices Accessed	Port Size	Size	Software Cache Inhibit	Notes
$00000000 - (DRAMSIZE -1)	User Programmable (Onboard DRAM)	D64	DRAMSIZE	N	1
DRAMSIZE - $FF7FFFFF	User Programmable (VMEbus)	D32/D16	3GB	?	2,3
$FF800000 - $FFBFFFFF	Flash Memory	D32	4MB	N	5
$FFC00000 - $FFEFFFFF	reserved	—	3MB	—	4
$FFF00000 - $FFFEFFFF	Local Devices (Refer to next table)	D32-D8	1MB	Y	—
$FFFF0000 - $FFFFFFFF	User Programmable (VMEbus A16)	D32/D16	64KB	?	1,3

Notes

1. This area is user-programmable. The suggested use is shown in the table. The DRAM decoder is programmed in the DCAM through the ECDM I^2C bus interface. The Processor Bus to Local Peripheral Bus and the Local Peripheral Bus to Processor Bus decoders are programmed in the BusSwitch. The Local Peripheral to VMEbus (master) and VMEbus to Local Peripheral Bus (slave) decoders are programmed in the VMEchip2.
2. Size is approximate.
3. Cache inhibit depends on devices in area mapped.
4. This area is not decoded. If these locations are accessed and the local peripheral bus timer is enabled, the cycle times out and is terminated by a TEA signal.
5. This area is user programmable via the BusSwitch. Default size is 4 megabytes.

The following table focuses on the Local Devices portion of the Memory Map.

Table 1-3. Local Devices Memory Map

Address Range	Devices Accessed	Port Size	Size	Notes
$FFF00000 - $FFF00FFF	BusSwitch	D64-D8	4KB	1
$FFF01000 - $FFF01FFF	ECDM (DCAM access)	---	4KB	1
$FFF02000 - $FFF02FFF	reserved	---	4KB	4
$FFF03000 - $FFF03FFF	reserved	---	4KB	4
$FFF04000 - $FFF04FFF	reserved	---	4KB	4
$FFF05000 - $FFF05FFF	reserved	---	4KB	4
$FFF06000 - $FFF06FFF	reserved	---	4KB	4
$FFF07000 - $FFF07FFF	User defined	---	4KB	4
$FFF08000 - $FFF3FFFF	reserved	---	224KB	4
$FFF40000 - $FFF400FF	VMEchip2 (LCSR)	D32	256B	1,2,3
$FFF40100 - $FFF401FF	VMEchip2 (GCSR)	D32-D8	256B	1,2,3
$FFF40200 - $FFF40FFF	reserved	---	3.5KB	4,5
$FFF41000 - $FFF41FFF	reserved	---	4KB	4
$FFF42000 - $FFF42FFF	PCCchip2	D32-D8	4KB	1,2
$FFF43000 - $FFF43FFF	reserved	---	4KB	4
$FFF44000 - $FFF44FFF	reserved	---	4KB	3
$FFF45000 - $FFF45FFF	CD2401 (Serial Comm. Cont.)	D16-D8	4KB	1,2
$FFF46000 - $FFF46FFF	82596CA (LAN)	D32	4KB	1,2,6
$FFF47000 - $FFF47FFF	53C710 (SCSI)	D32/D8	4KB	1,2
$FFF48000 - $FFF4FFFF	reserved	---	32KB	4
$FFF50000 - $FFF6FFFF	reserved	---	128KB	4
$FFF70000 - $FFF77FFF	reserved	---	32KB	4
$FFF78000 - $FFF7FFFF	reserved	---	288KB	4
$FFF80000 - $FFFBFFFF	DROM (BOOT ROM)	---	256KB	7
$FFFC0000 - $FFFCFFFF	MK48T08 (BBRAM,TOD Clk)	D32-D8	64KB	1,2
$FFFD0000 - $FFFEFFFF	reserved	---	128KB	4

Notes

1. For a complete description of the register bits, refer to the appropriate chapter or data sheet for the specific chip. For a more detailed memory map refer to the following detailed peripheral device memory maps.
2. Address is the physical address going to the device. It is after the BusSwitch translation from the MC88110 address to the device seen address.

3. **Writes to the LCSR in the VMEchip2 must be 32 bits. LCSR writes of 8 or 16 bits terminate with a TEA signal. Writes to the GCSR may be 8, 16, or 32 bits. Reads to the LCSR and GCSR may be 8, 16, or 32 bits.**
4. **This area does not return an acknowledge signal. If the processor bus timeout timer is enabled, the access times out and is terminated by a TEA signal.**
5. **Size is approximate.**
6. **Port commands to the 82596CA must be written as two 16-bit writes: upper word first and lower word second.**
7. **DROM (BOOT ROM) appears at $0 following a local peripheral bus reset. The DROM appears at 0 until the DR0 bit is cleared in the PCCchip2. In addition, the ROM0 bit in the ROMCR register of the BusSwitch must be cleared before the DRAM is accessed.**

Detailed I/O Memory Maps

Tables 1-4 through 1-15 give the detailed memory maps for the BusSwitch register, the ECDM CSR register, the DCAM (I^2C) register, the VMEchip2 register, the PCCchip2 register, the printer register, the CD2401 Serial Port register, the Ethernet LAN register, the SCSI Controller register, and the BBRAM/TOD Clock register.

Table 1-4. BusSwitch Register Memory Map

BusSwitch Base Address = $FFF00000

Offset	63 56	55 48	47	32	31	16	15	0
0	CHIPID	CHIPREV	GCSR		IODATA		IODIR	
8	PSAR1		PEAR1		PSAR2		PEAR2	
10	PSAR3		PEAR3		PSAR4		PEAR4	
18	PTR1		PTSR1		PTR2		PTSR2	
20	PTR3		PTSR3		PTR4		PTSR4	
28	SSAR1		SEAR1		SSAR2		SEAR2	
30	SSAR3		SEAR3		SSAR4		SEAR4	
38	STR1		STSR1		STR2		STSR2	
40	STR3		STSR3		STR4		STSR4	
48	PAR1	PAR2	PAR3	PAR4	SAR1	SAR2	SAR3	SAR4
50		BTIMER	PADJUST	PCOUNT	PAL			
58	WPPA				WPTPA		WPPAT	
60	ROMCR		TCTRL1	TCTRL2	LEVEL	MASK	ISEL0	ISEL1
68	ABORT	CPINT	TINT1	TINT2	WPINT	PALINT	XINT	VBASE
70	TCOMP1				TCOUNT1			
78	TCOMP2				TCOUNT2			
80	GPR1				GPR2			
88	GPR3				GPR4			
90	XCTAGS							
100	XCCR				VECTOR1			
108	VECTOR2				VECTOR3			
110	VECTOR4				VECTOR5			
118	VECTOR6				VECTOR7			

Base Address = $FF8XXXXX (MVME197DP and MVME197SP only)

XCFR

Table 1-5. ECDM CSR Register Memory Map

Sub-System Memory CSR Base Address = $FFF01000

Offset/Register:

ECDM0		ECDM1		ECDM2		ECDM3	
ADDR/REGISTER	ADDR/REGISTER	ADDR/REGISTER	ADDR/REGISTER	ADDR/REGISTER	ADDR/REGISTER	ADDR/REGISTER	ADDR/REGISTER
00 / MEMCON0	01 / ECDMID0	02 / MEMCON1	03 / ECDMID1	04 / MEMCON2	05 / ECDMID2	06 / MEMCON3	07 / ECDMID3
08 / SYNSTAT0	09 / ERSTAT0	0A / SYNSTAT1	0B / ERSTAT1	0C / SYNSTAT2	0D / ERSTAT2	0E / SYNSTAT3	0F / ERSTAT3
10 / I2CON0	11 / I2STAT0	12 / I2CON1	13 / I2STAT1	14 / I2CON2	15 / I2STAT2	16 / I2CON3	17 / I2STAT3
18 / I2DATA0	19 / I2ADDR0	1A / I2DATA1	1B / I2ADDR1	1C / I2DATA2	1D / I2ADDR2	1E / I2DATA3	1F / I2ADDR3

D63 D56	D55 D48	D47 D40	D39 D32	D31 D24	D23 D16	D15 D8	D7 D0

ECDM register map of four ECDM devices in a 64-bit system. The byte offset address is shown next to each register.

Table 1-6. DCAM (I^{2}C) Register Memory Map

DCAM (I^{2}C) Base Address = $C0 (default)

Offset

	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
00 00	ID Register							
01 01	Version Register							
02 02	SL31	SL30	SL29	SL28	SL27	SL26	SL25	DISRAM
03 03	SH31	SH30	SH29	SH28	SH27	SH26	SH25	SCRUB1TIME
04 04	CASCLKSL	CASCLK2	CASCLK1	PGMODE	ONEBANK	DRAMSIZ3	DRAMSIZ2	DRAMSIZ1
05 05	REF7	REF6	REF5	REF4	REF3	REF2	REF1	REF0
06 06	REFTAIL4	REFTAIL3	REFTAIL2	REFTAIL1	REF11	REF10	REF9	REF8
07 07	NOT USED	NOT USED	RDTAIL5	RDTAIL4	RDTAIL3	RDTAIL2	RDTAIL1	RTCLKSL
08 08	READACK7	READACK6	READACK5	READACK4	READACK3	READACK2	READACK1	INTRRUPT
09 09	NOT USED	READOE6	READOE5	READOE4	READOE3	READOE2	READOE1	NOT USED
0A 10	FECCKSL	BREADOE6	BREADOE5	BREADOE4	BREADOE3	BREADOE2	BREADOE1	PCGCLKSL
0B 11	PCHG7	PCHG6	PCHG5	PCHG4	PCHG3	PCHG2	PCHG1	PCHG0
0C 12	SLECDM5	SLECDM4	SLECDM3	SLECDM2	FLECDM4	FLECDM3	FLECDM2	FLECDM1
0D 13	NOT USED	ERAMOE6	ERAMOE5	ERAMOE4	ERAMOE3	ERAMOE2	ERAMOE1	ROECLKSL
0D 14	NOT USED	RMWRMOE6	RMWRMOE5	RMWRMOE4	RMWRMOE3	RMWRMOE2	RMWRMOE1	RMWOE5
0F 15	CSRTAIL7	CSRTAIL6	CSRTAIL5	CSRTAIL4	CSRTAIL3	CSRTAIL2	CSRTAIL1	NOT USED
10 16	BWRTTL4	BWRTTL3	BWRTTL2	BWRTTL1	RMWOE4	RMWOE3	RMWOE2	RMWOE1
11 17	SECCLKSL	RMWOCKSL	BWRITE5	BWRITE4	BWRITE3	BWRITE2	BWRITE1	WRCLKSEL
12 18	NOT USED	NOT USED	RMW5	RMW4	RMW3	RMW2	RMW1	NOT USED
13 19	RMWTAIL7	RMWTAIL6	RMWTAIL5	RMWTAIL4	RMWTAIL3	RMWTAIL2	RMWTAIL1	RMWTLCSL
14 20	CBRDOE3	CBRDOE2	CBRDOE1	NOT USED	CREADOE3	CREADOE2	CREADOE1	BWRTCSL
15 21	SC9	SC8	SC7	SC6	SC5	SC4	SC3	SC2
16 22	SC17	SC16	SC15	SC14	SC13	SC12	SC11	SC10
17 23	SC25	SC24	SC23	SC22	SC21	SC20	SC19	SC18
18 24	NOT USED	SC32	SC31	SC30	SC29	SC28	SC27	SC26
19 25	NOT USED	NOT USED	NOT USED	CBTAIL4	CBTAIL3	CBTAIL2	CBTAIL1	CBTLCKSL
1A 26	CSR7	CSR6	CSR5	CSR4	NOT USED	NOT USED	NOT USED	NOT USED
1B 27	CSR15	CSR14	CSR13	CSR12	CSR11	CSR10	CSR9	CSR8
1C 28	CSR23	CSR22	CSR21	CSR20	CSR19	CSR18	CSR17	CSR16
1D 29	CSR31	CSR30	CSR29	CSR28	CSR27	CSR26	CSR25	CSR24
1E 30	NOT USED	NOT USED	BRDTAIL5	BRDTAIL4	BRDTAIL3	BRDTAIL2	BRDTAIL1	NOT USED
1F 31								
	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0

DCAM registers are only accessible/addressable on the DRAM sub-system I^{2}C bus through the ECDM I^{2}C interface.

Table 1-7. VMEchip2 Memory Map (Sheet 1 of 4)

VMEchip2 LCSR Base Address = $FFF40000
OFFSET:

	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
00	VMEbus SLAVE ENDING ADDRESS 1																VMEbus SLAVE STARTING ADDRESS 1															
04	VMEbus SLAVE ENDING ADDRESS 2																VMEbus SLAVE STARTING ADDRESS 2															
08	VMEbus SLAVE ADDRESS TRANSLATION ADDRESS 1																VMEbus SLAVE ADDRESS TRANSLATION SELECT 1															
0C	VMEbus SLAVE ADDRESS TRANSLATION ADDRESS 2																VMEbus SLAVE ADDRESS TRANSLATION SELECT 2															
10						(VB) SNP 2		(VB) WP 2	(VB) SUP 2	(VB) USR 2	(VB) A32 2	(VB) A24 2	(VB) D64 2	(VB) BLK 2	(VB) PGM 2	(VB) DAT 2						(VB) SNP 1		(VB) WP 1	(VB) SJP 1	(VB) USR 1	(VB) A32 1	(VB) A24 1	(VB) D64 1	(VB) BLK 1	(VB) PGM 1	(VB) DAT 1
14	LOCAL BUS SLAVE ENDING ADDRESS 1																LOCAL BUS SLAVE STARTING ADDRESS 1															
18	LOCAL BUS SLAVE ENDING ADDRESS 2																LOCAL BUS SLAVE STARTING ADDRESS 2															
1C	LOCAL BUS SLAVE ENDING ADDRESS 3																LOCAL BUS SLAVE STARTING ADDRESS 3															
20	LOCAL BUS SLAVE ENDING ADDRESS 4																LOCAL BUS SLAVE STARTING ADDRESS 4															
24	LOCAL BUS SLAVE ADDRESS TRANSLATION ADDRESS 4																LOCAL BUS SLAVE ADDRESS TRANSLATION SELECT 4															
28	(LB) D16 EN	(LB) WP EN	(LB) AM 4						(LB) D16 EN	(LB) WP EN	(LB) AM 3						(LB) D16 EN	(LB) WP EN	(LB) AM 2						(LB) D16 EN	(LB) WP EN	(LB) AM 1					
2C	(VB) GCSR GROUP ADDRESS								(VB) GCSR BOARD ADDRESS				LB EN4	LB EN3	LB EN2	LB EN1	LB I2 EN	LB I2 WP	LB I2 SU	LB I2 PD	LB I1 EN	LB I1 D16	LB I1 WP	LB I1 SU	ROM SIZE (XX)		ROM BANK B SPEED (XX)			ROM BANK A SPEED (XX)		
	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

LB	= Local Bus		VB	= VMEbus
(LB)	= Local Bus Slave		(VB)	= VMEbus Slave
LV	= Local Bus to VMEbus		(XX)	= Not Used on the MVME197 Series

Table 1-7. VMEchip2 Memory Map (Continued) (Sheet 2 of 4)

VMEchip2 LCSR Base Address = $FFF40000
OFFSET:

	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
30												ROMO (XX)	DMAC TB SNP MODE (XX)		SPRAM SPEED (XX)		LV ROBN MODE	LV DHB	LV DWB		LV FAIR	LV RWD	LV REQUEST LEVEL		DMAC HALT	DMAC EN	DMAC TBL	DMAC FAIR	DMAC RELM		DMAC REQUEST LEVEL	
34																	DMAC INTE	DMAC LB SNP MODE			DMAC VME INC	DMAC LB INC	DMAC TVME	DMAC D16	DMAC BLK TR	DMAC BLK TR	DMAC AM	DMAC AM	DMAC AM	DMAC AM	DMAC AM	DMAC AM
38	DMAC LOCAL BUS ADDRESS COUNTER																															
3C	DMAC VMEbus ADDRESS COUNTER																															
40	DMAC BYTE COUNTER																															
44	DMAC TABLE ADDRESS COUNTER																															
48		INTERRUPT 1 SIGNAL		IRQ CLR	IRQ STAT	VMEbus INTERRUPT LEVEL			VMEbus INTERRUPT VECTOR								DMAC INTERRUPT COUNTER				MPU CLR	MPU LB EN	MPU LB PERR	MPU LB OB	MPU LB TO	DMAC LB EN	DMAC LB PERR	DMAC LB OB	DMAC LB TO	DMAC LB ERR	DMAC VME ERR	DMAC DONE
4C								VB TO	DMAC TIME OFF			DMAC TIME ON			VMEbus GLOBAL TIMEOUT		VMEbus ACCESS TIMEOUT		LOCAL BUS TIMEOUT		WATCHDOG TIMEOUT				PRESCALER ADJUST							
50	TICK TIMER 1 COMPARE																															
54	TICK TIMER 1 COUNTER																															
58	TICK TIMER 2 COMPARE																															
5C	TICK TIMER 2 COUNTER																															
60		SCON	SYS FAIL	BRD FAIL STAT	PURS STAT	CLR PURS STAT	BRD FAIL OUT	RST SW EN	SYS RST	WD CLR STAT	WD CLR CNT	WD TO STAT	WD BDF EN	WD SYS RST	WD RST EN	WD EN	OVERFLOW COUNTER 2					CLR OVF 2	COC 2	CNT EN 2	OVERFLOW COUNTER 1					CLR OVF 1	COC 1	CNT EN 1
64	PRESCALER COUNTER																															
	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

LB	= Local Bus	VB	= VMEbus
(LB)	= Local Bus Slave	(VB)	= VMEbus Slave
LV	= Local Bus to VMEbus	(XX)	= Not Used on the MVME197 Series

Table 1-7. VMEchip2 Memory Map (Continued) (Sheet 3 of 4)

VMEchip2 LCSR Base Address = $FFF40000

OFFSET:

	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
68	AC FAIL IRQ	AB SW IRQ	SYS FAIL IRQ	MWP ERR IRQ	PE IRQ	IRQ1 EDGE IRQ	TIC TIM2 IRQ	TIC TIM1 IRQ	VME ACK IRQ	DMAC IRQ	GCSR SIG3 IRQ	GCSR SIG2 IRQ	GCSR SIG1 IRQ	GCSR SIG0 IRQ	GCSR LM1 IRQ	GCSR LM0 IRQ	LB SW7 IRQ	LB SW6 IRQ	LB SW5 IRQ	LB SW4 IRQ	LB SW3 IRQ	LB SW2 IRQ	LB SW1 IRQ	LB SW0 IRQ	SPARE	VME IRQ7 IRQ	VME IRQ6 IRQ	VME IRQ5 IRQ	VME IRQ4 IRQ	VME IRQ3 IRQ	VME IRQ2 IRQ	VME IRQ1 IRQ
6C	EN IRQ 31	EN IRQ 30	EN IRQ 29	EN IRQ 28	EN IRQ 27	EN IRQ 26	EN IRQ 25	EN IRQ 24	EN IRQ 23	EN IRQ 22	EN IRQ 21	EN IRQ 20	EN IRQ 19	EN IRQ 18	EN IRQ 17	EN IRQ 16	EN IRQ 15	EN IRQ 14	EN IRQ 13	EN IRQ 12	EN IRQ 11	EN IRQ 10	EN IRQ 9	EN IRQ 8	EN IRQ 7	EN IRQ 6	EN IRQ 5	EN IRQ 4	EN IRQ 3	EN IRQ 2	EN IRQ 1	EN IRQ 0
70																	SET IRQ 15	SET IRQ 14	SET IRQ 13	SET IRQ 12	SET IRQ 11	SET IRQ 10	SET IRQ 9	SET IRQ 8								
74	CLR IRQ 31	CLR IRQ 30	CLR IRQ 29	CLR IRQ 28	CLR IRQ 27	CLR IRQ 26	CLR IRQ 25	CLR IRQ 24	CLR IRQ 23	CLR IRQ 22	CLR IRQ 21	CLR IRQ 20	CLR IRQ 19	CLR IRQ 18	CLR IRQ 17	CLR IRQ 16	CLR IRQ 15	CLR IRQ 14	CLR IRQ 13	CLR IRQ 12	CLR IRQ 11	CLR IRQ 10	CLR IRQ 9	CLR IRQ 8								
78		ACFAIL IRQ LEVEL				ABORT IRQ LEVEL				SYSFAIL IRQ LEVEL				MASTER WRITE POST ERROR IRQ LEVEL				PARITY ERROR IRQ LEVEL				IRQ1 EDGE-SENSITIVE IRQ LEVEL				TICK TIMER 2 IRQ LEVEL				TICK TIMER 1 IRQ LEVEL		
7C		VMEbus ACKNOWLEDGE IRQ LEVEL				DMAC IRQ LEVEL				GCSR SIG 3 IRQ LEVEL				GCSR SIG 2 IRQ LEVEL				GCSR SIG 1 IRQ LEVEL				GCSR SIG 0 IRQ LEVEL				GCSR LM 1 IRQ LEVEL				GCSR LM 0 IRQ LEVEL		
80		SW7 IRQ LEVEL				SW6 IRQ LEVEL				SW5 IRQ LEVEL				SW4 IRQ LEVEL				SW3 IRQ LEVEL				SW2 IRQ LEVEL				SW1 IRQ LEVEL				SW0 IRQ LEVEL		
84		SPARE IRQ LEVEL				VMEbus IRQ7 IRQ LEVEL				VMEbus IRQ6 IRQ LEVEL				VMEbus IRQ5 IRQ LEVEL				VMEbus IRQ4 IRQ LEVEL				VMEbus IRQ3 IRQ LEVEL				VMEbus IRQ2 IRQ LEVEL				VMEbus IRQ1 IRQ LEVEL		
88	VECTOR BASE REGISTER 0				VECTOR BASE REGISTER 1				MST IRQ EN	SYS FAIL LEVEL	AC FAIL LEVEL	ABORT LEVEL	GENERAL PURPOSE I/O ENABLE				GENERAL PURPOSE I/O OUTPUT				GENERAL PURPOSE I/O INPUT				GENERAL PURPOSE INPUT							
	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

LB	= Local Bus	VB	= VMEbus
(LB)	= Local Bus Slave	(VB)	= VMEbus Slave
LV	= Local Bus to VMEbus	(XX)	= Not Used on the MVME197 Series

Table 1-7. VMEchip2 Memory Map (Continued) (Sheet 4 of 4)

VMEchip2 GCSR Base Address = $FFF40100

L	V	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0	0	CHIP REVISION								CHIP ID							
4	2	LM3	LM2	LM1	LM0	SIG3	SIG2	SIG1	SIG0	RST	ISF	BF	SCON	SYS FL			
8	4	GENERAL PURPOSE CONTROL AND STATUS REGISTER 0															
C	6	GENERAL PURPOSE CONTROL AND STATUS REGISTER 1															
10	8	GENERAL PURPOSE CONTROL AND STATUS REGISTER 2															
14	A	GENERAL PURPOSE CONTROL AND STATUS REGISTER 3															
18	C	GENERAL PURPOSE CONTROL AND STATUS REGISTER 4															
1C	E	GENERAL PURPOSE CONTROL AND STATUS REGISTER 5															
		D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

NOTES: L = Local Bus Offset.
V = VMEbus Offset.

Table 1-8. PCCchip2 Memory Map

PCCchip2 LCSR Base Address = $FFF42000

OFFSET:

	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
00	CHIP ID								CHIP REVISION								DR0					CPU 040	MSTR INT EN	FAST BRAM	VECTOR BASE							
04	TIC TIMER 1 COMPARE																															
08	TIC TIMER 1 COUNTER																															
0C	TIC TIMER 2 COMPARE																															
10	TIC TIMER 2 COUNTER																															
14	PRESCALER COUNT								PRE SCALER CLOCK ADJUST								OVERFLOW COUNTER 2					CLR OVF 2	COC 2	TIC 2	OVERFLOW COUNTER 1					CLR OVF 1	COC 1	TIC 1
18	GPI PLTY	GPI E/L*	GPI INT	GPI IEN	GPI ICLR	GPI IRQ LEVEL								GPI	GIOE	GPO			TIC2 INT	TIC2 IEN	TIC2 ICLR	TIC TIMER 2 IRQ LEVEL					TIC1 INT	TIC1 IEN	TIC1 ICLR	TIC TIMER 1 IRQ LEVEL		
1C				SCC RTRY ERR	SCC PAR ERR	SCC EXT ERR	SCC LTO ERR	SCC SCLR			SCC MDM	SCC MDM	SCC MDM	SCC MODEM IRQ LEVEL					SCC TX IRQ	SCC TX IEN	SCC TX AVEC	SCC TRANSMIT IRQ LEVEL			SCC SC1	SCC SC2	SCC IRQ	SCC IEN	SCC AVEC	SCC RECEIVE IRQ LEVEL		
20																									SCC MODEM PIACK							
24									SCC TRANSMIT PIACK																SCC RECEIVE PIACK							
28					LAN PAR ERR	LAN EXT ERR	LAN LTO ERR	LAN SCLR									LAN PLTY	LAN E/L*	LAN INT	LAN IEN	LAN ICLR	LAN IRQ LEVEL			LAN SC1	LAN SC2	LAN ERR INT	LAN ERR IEN	LAN ERR ICLR	LAN ERR IRQ LEVEL		
2C					SCSI PAR ERR	SCSI EXT ERR	SCSI LTO ERR	SCSI SCLR																			SCSI IRQ	SCSI IEN		SCSI INT IRQ LEVEL		
30	PRTR ACK PLTY	PRTR ACK E/L*	PRTR ACK INT	PRTR ACK IEN	PRTR ACK ICLR	PRTR ACK IRQ LEVEL			PRTR FLT PLTY	PRTR FLT E/L*	PRTR FLT INT	PRTR FLT IEN	PRTR FLT ICLR	PRTR FAULT IRQ LEVEL			PRTR SEL PLTY	PRTR SEL E/L*	PRTR SEL INT	PRTR SEL IEN	PRTR SEL ICLR	PRTR SEL IRQ LEVEL			PRTR PE PLTY	PRTR PE E/L*	PRTR PE INT	PRTR PE IEN	PRTR PE ICLR	PRTR PE IRQ LEVEL		
34	PRTR BSY PLTY	PRTR BSY E/L*	PRTR BSY INT	PRTR BSY IEN	PRTR BSY ICLR	PRTR BSY IRQ LEVEL											PRTR ANY INT			PRTR ACK	PRTR FLT	PRTR SEL	PRTR PE	PRTR BSY				PRTR DAT ENBL	PRTR INP	PRTR STB	PRTR FAST ASTB	PRTR MAN STB
38	CHIP SPEED																PRINTER DATA															
3C																						INTERRUPT PL LEVEL								INTERRUPT MASK LEVEL		
	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Table 1-9. Printer Memory Map

Printer ACK Interrupt Control Register $FFF42030

BIT	31	30	29	28	27	26	25	24
NAME	PLTY	E/L*	INT	IEN	ICLR	IL2	IL1	IL0

Printer FAULT Interrupt Control Register $FFF42031

BIT	23	22	21	0	19	18	17	16
NAME	PLTY	E/L*	INT	IEN	ICLR	IL2	IL1	IL0

Printer SEL Interrupt Control Register $FFF42032

BIT	15	14	13	12	11	10	9	8
NAME	PLTY	E/L*	INT	IEN	ICLR	IL2	IL1	IL0

Printer PE Interrupt Control Register $FFF42033

BIT	7	6	5	4	3	2	1	0
NAME	PLTY	E/L*	INT	IEN	ICLR	IL2	IL1	IL0

Printer BUSY Interrupt Control Register $FFF42034

BIT	31	30	29	28	27	26	25	24
NAME	PLTY	E/L*	INT	IEN	ICLR	IL2	IL1	IL0

Printer Input Status Register $FFF42036

BIT	15	14	13	12	11	10	9	8
NAME	PINT			ACK	FLT	SEL	PE	BSY

Printer Port Control Register $FFF42037

BIT	7	6	5	4	3	2	1	0
NAME				DOEN	INP	STB	FAST	MAN

Printer Data Register (16 bits) $FFF4203A

BIT	15-0
NAME	PD15-PD0

Printer memory map is part of the PCCchip2 (refer to PCCchip2 Memory Map).

Table 1-10. Cirrus Logic CD2401 Serial Port Memory Map

Cirrus Logic CD2400 Memory Map		Base Address Is $FFF45000		
		Offsets	Size	Access
Global Firmware Revision Code Register	(GFRCR)	81	B	R
Transmit FIFO Transfer Count	(TFTC)	80	B	R
Modem End Of Interrupt Register	(MEOIR)	86	B	R/W
Transmit End Of Interrupt Register	(TEOIR)	85	B	R/W
Receive End Of Interrupt Register	(REOIR)	84	B	R/W
Modem (/Timer) Interrupt Status Register	(MISR)	8B	B	R
Transmit Interrupt Status Register	(TISR)	8A	B	R
Receive Interrupt Status Register	(RISR)	88	W (NOTE)	R
Receive Interrupt Status Register low	(RISRl)	89	B	R
Receive Interrupt Status Register high	(RISRh)	88	B	R
Timer Period Register	(TPR)	DA	B	R/W
Priority Interrupt level Register 1	(PILR1)	E3	B	R/W
Priority Interrupt level Register 2	(PILR2)	E0	B	R/W
Priority Interrupt level Register 3	(PILR3)	E1	B	R/W
Channel Access Register	(CAR)	EE	B	R/W
Receive Data Register	(RDR)	F8	B	R
Transmit Data Register	(TDR)	F8	B	W
Local Interrupting Channel Register	(LICR)	26	B	R/W
Local Interrupt Vector Register	(LIVR)	09	B	R/W
Channel Command Register	(CRR)	13	B	R/W
Special Transmit Command Register	(STCR)	12	B	R/W
Interrupt Enable Register	(IER)	11	B	R/W
Channel Option Register 1	(COR1)	10	B	R/W
Channel Option Register 2	(COR2)	17	B	R/W
Channel Option Register 3	(COR3)	16	B	R/W
Channel Option Register 4	(COR4)	15	B	R/W
Channel Option Register 5	(COR5)	14	B	R/W
Channel Mode Register	(CMR)	1B	B	R/W

This is a 16-bit register.

Table 1-11. 82596CA Ethernet LAN Memory Map

**82596CA Ethernet LAN
Directly Accessible Registers**

Address	Data Bits	
	D31 D16	D15 D0
$FFF46000	Upper Command Word	Lower Command Word
$FFF46004	MPU Channel Attention (CA)	

Notes

1. **Refer to the MPU Port and MPU Channel Attention registers in the *PCCchip2* chapter.**
2. **After reset you must write the System Configuration Pointer to the command registers prior to writing to the CPU Channel Attention register. Writes to the System Configuration Pointer must be upper word first, lower word second.**

Table 1-12. 53C710 SCSI Memory Map

53C710 Register Address Map			Base Address is $FFF47000		
Big Endian Mode					SCRIPTs Mode and Little Endian Mode
00	SIEN	SDID	SCNTL1	SCNTL0	00
04	SOCL	SODL	SXFER	SCID	04
08	SBCL	SBDL	SIDL	SFBR	08
0C	SSTAT2	SSTAT1	SSTAT0	DSTAT	0C
10	DSA				10
14	CTEST3	CTEST2	CTEST1	CTEST0	14
18	CTEST7	CTEST6	CTEST5	CTEST4	18
1C	TEMP				1C
20	LCRC	CTEST8	ISTAT	DFIFO	20
24	DCMD	DBC			24
28	DNAD				28
2C	DSP				2C
30	DSPS				30
34	SCRATCH				34
38	DCNTL	DWT	DIEN	DMODE	38
3C	ADDER				3C

Note Accesses may be 8-bit or 32-bit, but not 16-bit.

Table 1-13. MK48T08 BBRAM, TOD Clock Memory Map

Address Range	Description	Size (Bytes)
$FFFC0000 - $FFFC0FFF	User Area	4096
$FFFC1000 - $FFFC10FF	Networking Area	256
$FFFC1100 - $FFFC16F7	Operating System Area	1528
$FFFC16F8 - $FFFC1EF7	Debugger Area	2048
$FFFC1EF8 - $FFFC1FF7	Configuration Area	256
$FFFC1FF8 - $FFFC1FFF	TOD Clock	8

Table 1-14. BBRAM Configuration Area Memory Map

Address Range	Description	Size (Bytes)
$FFFC1EF8 - $FFFC1EFB	Version	4
$FFFC1EFC - $FFFC1F07	Serial Number	12
$FFFC1F08 - $FFFC1F17	Board ID	16
$FFFC1F18 - $FFFC1F27	PWA	16
$FFFC1F28 - $FFFC1F2B	Speed	4
$FFFC1F2C - $FFFC1F33	Ethernet Address	8
$FFFC1F34 - $FFFC1FF6	Reserved	195
$FFFC1FF7	Checksum	1

Table 1-15. TOD Clock Memory Map

Address	Data Bits								Function
	D7	D6	D5	D4	D3	D2	D1	D0	
$FFFC1FF8	W	R	S	--	--	--	--	--	CONTROL
$FFFC1FF9	ST	--	--	--	--	--	--	--	SECONDS 00
$FFFC1FFA	x	--	--	--	--	--	--	--	MINUTES 00
$FFFC1FFB	x	x	--	--	--	--	--	--	HOUR 00
$FFFC1FFC	x	x	x	x	x	--	--	--	DAY 01
$FFFC1FFD	x	x	FT	--	--	--	--	--	DATE 01
$FFFC1FFE	x	x	x	--	--	--	--	--	MONTH 01
$FFFC1FFF	--	--	--	--	--	--	--	--	YEAR 00

Notes

ST	**=**	**Stop Bit**	**S**	**=**	**Sign Bit**
W	**=**	**Write Bit**	**FT**	**=**	**Frequency Test**
R	**=**	**Read Bit**	**x**	**=**	**Unused**
--	**=**	**Data Bit**			

BBRAM, TOD Clock Memory Map

The MK48T08 BBRAM (also called Non-Volatile RAM or NVRAM) is divided into six areas as shown in Table 1-13. The first five areas are defined by software, while the sixth area, the time-of-day (TOD) clock, is defined by the chip hardware. The first area is reserved for user data. The second area is used by Motorola networking software. The third area is used by the SYSTEM V/

88 operating system. The fourth area is used by the MVME197 board debugger. The fifth area, detailed in Table 1-14, is the configuration area. The sixth area, the TOD clock, detailed in Table 1-15, is defined by the chip hardware.

The data structure of the configuration bytes starts at $FFFC1EF8 and is as follows.

```
struct config_rom {
            char        version[4];
            char        serial[12];
            char        id[16];
            char        pwa[16];
            char        speed[4];
            char        ethernet_adr[8];
            char        reserved[195];
            char        cksum[1];
}
```

The fields are defined as follows:

1. Four bytes are reserved for the revision or version of this structure. This revision is stored in ASCII format, with the first two bytes being the major version numbers and the last two bytes being the minor version numbers. For example, if the version of a structure is 4.6, this field contains:

   0460

2. Twelve bytes are reserved for the serial number of the board in ASCII format. For example, this field could contain:

   000000470476

3. Sixteen bytes are reserved for the board ID in ASCII format. For example, for a MVME197LE module, this field contains:

   MVME197LE

   (The nine characters are followed by seven blanks.)

4. Sixteen bytes are reserved for the printed wiring assembly (PWA) number assigned to this board in ASCII format. This includes the 01-W prefix. This is for the main logic board if more than one board is required for a set. Additional boards in a set are defined by a structure for that set. For example, for a 32 megabyte, 50 MHz MVME197LE board at revision A, the PWA field contains:

01-W3869B03A

(The 12 characters are followed by four blanks.)

5. Four bytes contain the speed of the board in MHz. The first two bytes are the whole number of MHz and the second two bytes are fractions of MHz. For example, for a 50 MHz board, this field contains:

   5000

6. Eight bytes are reserved for the Ethernet address. The address is stored in hexadecimal format, with the last two bytes not used. (Refer to the *Ethernet Interface* section for a more detailed description). If the board does not support Ethernet, this field is filled with zeros.
7. Growth space (195 bytes) is reserved. This pads the structure to an even 256 bytes. Board-specific items, such as mezzanine board PWA numbers, may go here.
8. The final one byte of the area is reserved for a checksum (as defined in the *MVME197BUG 197Bug Debugging Package User's Manual*) for security and data integrity of the configuration area of the NVRAM. This data is stored in hexadecimal format.

VMEbus Memory Map

This section describes the mapping of local resources as viewed by VMEbus masters.

VMEbus Accesses to the Local Peripheral Bus

The VMEchip2 includes a user-programmable map decoder for the VMEbus to local peripheral bus interface. The map decoder allows the user to program the starting and ending address and the modifiers the MVME197 responds to.

VMEbus Short I/O Memory Map

The VMEchip2 includes a user-programmable map decoder for the GCSR (Global Control and Status Registers). The GCSR map decoder allows the user to program the starting address of the GCSR in the VMEbus short I/O space.

Software Support Considerations

The MVME197 is a complex board that interfaces to the VMEbus and SCSI bus. These two bus interfaces raise the issue of cache coherency and support of indivisible cycles. There are also many sources of bus error.

VMEchip2 2

Introduction

This chapter defines the VMEchip2, local peripheral bus to VMEbus interface chip.

The VMEchip2 interfaces the local peripheral bus to the VMEbus. In addition to the VMEbus defined functions, the VMEchip2 includes a local peripheral bus to VMEbus DMA controller, VME board support features, and Global Control and Status Registers (GCSR) for interprocessor communications.

VMEchip2 Features

- □ Local Peripheral Bus to VMEbus Interface
  - Programmable local peripheral bus map decoder
  - Programmable short, standard and extended VMEbus addressing
  - Programmable AM codes
  - Programmable 16-bit and 32-bit VMEbus data width
  - Software-enabled write posting mode
  - Write post buffer (one cache line or one four-byte)
  - Automatically performs dynamic bus sizing for VMEbus cycles
  - Software-configured VMEbus access timers
  - Local peripheral bus to VMEbus Requester
    - Software-enabled FAIR request mode
    - Software-configured release modes
      - Release-When-Done (RWD)
      - Release-On-Request (ROR)
    - Software-configured BR0*-BR3* request levels
- □ VMEbus to Local Peripheral Bus Interface
  - Programmable VMEbus map decoder
  - Programmable AM decoder
  - Programmable local peripheral bus snoop enable (This feature is not applicable to any of the MVME197 module series)
  - Simple VMEbus to local peripheral bus address translation

  - 8-bit, 16-bit and 32-bit VMEbus data width
  - 8-bit, 16-bit and 32-bit block transfer
  - Standard and extended VMEbus addressing
  - Software-enabled write posting mode
  - Write post buffer (9 four-bytes in BLT mode, 2 four-bytes in non-BLT mode)
  - An eight four-byte read ahead buffer (BLT mode only)
- ❑ 32-Bit Local - VMEbus DMA Controller
  - Programmable 16-bit, 32-bit, and 64-bit VMEbus data width
  - Programmable short, standard and extended VMEbus addressing
  - Programmable AM code
  - Programmable local peripheral bus snoop enable (This feature is not applicable to the MVME197 module series)
  - A 16 four-byte FIFO data buffer
  - Supports up to 4 gigabytes of data per DMA request
  - Automatically adjusts transfer size to optimize bus utilization
  - DMA complete interrupt
  - DMAC command chaining is supported by a singly-linked list of DMA commands
  - VMEbus DMA controller requester
    - Software-enabled FAIR request modes
    - Software-configured release modes
      - Release-On-Request (ROR)
      - Release-On-End-Of-Data (ROEOD)
    - Software-configured BR0-BR3 request levels
    - Software enabled bus-tenure timer
- ❑ VMEbus Interrupter
  - Software-configured IRQ1-IRQ7 interrupt request level
  - 8-bit software-programmed status/ID register
- ❑ VMEbus System Controller
  - Arbiter with software-configured arbitration modes
    - Priority (PRI)

  - Round-Robin-Select (RRS)
  - Single-level (SGL)
- Programmable arbitration timer
- IACK daisy-chain driver
- Programmable bus timer
- SYSRESET logic

❑ Global Control Status Register Set

- Four location monitors
- Global control of locally detected failures
- Global control of local reset
- Four global attention interrupt bits
- A chip ID and revision register
- Four 16-bit dual-ported general purpose registers

❑ Interrupt Handler

- All interrupts are level-programmable
- All interrupts are maskable
- All interrupts provide a unique vector
- Software and external interrupts

❑ Watchdog timer

❑ Two 32-bit tick timers

❑ Map decoder and control for two banks of EPROM (This feature is not applicable to any of the MVME197 module series)

❑ Map decoder and control for one bank of static RAM

❑ Support for RESET and ABORT switches

Functional Description

The following sections provide an overview of the functions provided by the VMEchip2. See Figure 2-1 for a block diagram of the VMEchip2. A detailed programming model for the local control and status registers (LCSR) is provided in the following section. A detailed programming model for the global control and status registers (GCSR) is provided in the following section.

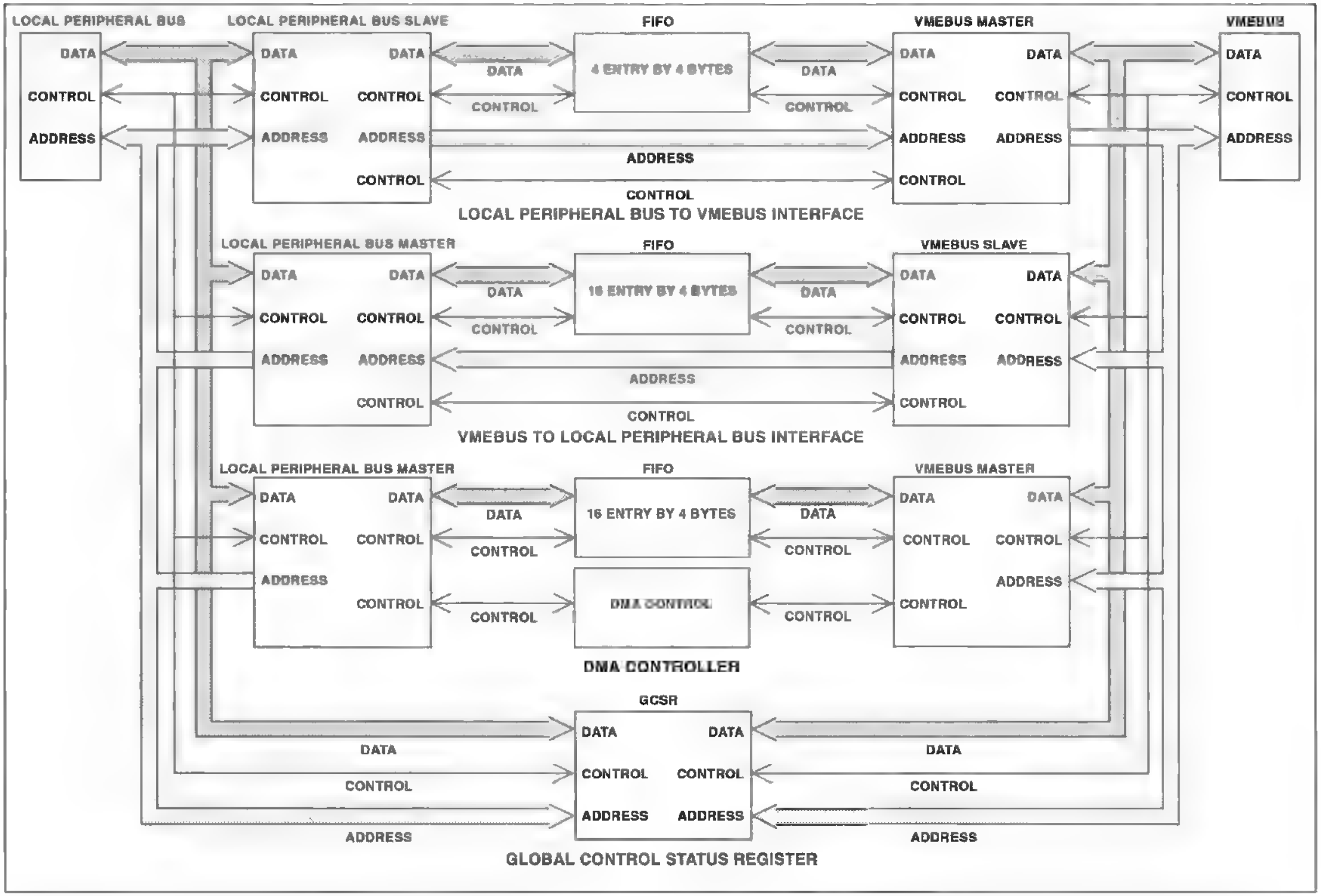


Figure 2-1. VMEchip2 Block Diagram

Local Peripheral Bus to VMEbus Interface

The local peripheral bus to VMEbus interface allows local peripheral bus masters access to global resources on the VMEbus. This interface includes a local peripheral bus slave, a write post buffer and a VMEbus master.

Using programmable map decoders with programmable attribute bits, the local peripheral bus to VMEbus interface can be configured to provide the following VMEbus capabilities:

Addressing capabilities:	A16, A24, A32
Data transfer capabilities:	D08, D16, D32

The local peripheral bus slave includes five programmable local peripheral bus map decoders for accessing the VMEbus. The first four map decoders are general purpose while the fifth decoder is dedicated for short I/O decoding. The first four map decoders compare local peripheral bus address lines A31 through A16 with a 16-bit start address and a 16-bit end address. When an address in the selected range is detected, a VMEbus select is generated to the VMEbus master. Each map decoder also has eight attribute bits and an enable bit. The attribute bits are for VMEbus AM codes, D16 enable, and write post (WP) enable.

The fourth map decoder also includes a 16-bit alternate address register and a 16-bit alternate address select register. This allows any or all of the upper 16 address bits from the local peripheral bus to be replaced by bits from the alternate address register. The feature allows the local peripheral bus master to access any VMEbus address.

Using the four map decoders, separate VMEbus maps can be created, each with its own attributes. For example, one map can be configured as A32, D32 with write posting enabled while a second map can be A24, D16 with write posting disabled.

The fifth map decoder decodes local peripheral bus addresses $FFFF0000 through $FFFFFFFF as the short I/O A16 area. The VMEbus AM code is always A16. Supervisor/non-privileged and program/data space is determined by the local peripheral bus function codes. Write posting may be enabled or disabled for the short I/O space and this map decoder may be enabled or disabled.

When write posting is enabled, the VMEchip2 stores the local peripheral bus address and data and then acknowledges the local peripheral bus master. The local peripheral bus is then free to perform other operations while the VMEbus master requests the VMEbus and performs the requested operation.

The write post buffer stores one byte, two-byte, four-byte or one cache line four 4-bytes). Write posting should only be enabled when bus errors are not expected. If a bus error is returned on a write posted cycle, the local master is interrupted. The address of the error is not saved. Normal memory never returns a bus error on a write cycle. However, some ECC memory cards perform a read-modify-write operation and therefore may return a bus error if there is an error on the read portion of a read-modify-write. Write posting should not be enabled when this type of memory card is used. Also, memory should not be sized using write operations if write posting is enabled. I/O areas that have holes should not be write posted if software may access non-existent memory Using the programmable map decoders, write posting can be enabled for "safe" areas and disabled for areas which are not "safe".

Using programmable map decoders with programmable attribute bits, the local peripheral bus to VMEbus interface can be configured to provide the following VMEbus capabilities:

Addressing capabilities:	A16, A24, A32
Data transfer capabilities:	D08, D16, D32

Block transfer is not supported because the MC68040 block transfer capability is not compatible with the VMEbus.

The VMEbus master supports dynamic bus sizing. When a device on the local peripheral bus initiates a quad-byte access to a VMEbus slave that only has the D16 data transfer capability, the chip executes two double-byte cycles on the VMEbus, acknowledging the local device after all requested 4-bytes have been accessed. This enhances the portability of software because it allows software to run on the system regardless of the physical organization of global memory.

Using the local peripheral bus map decoder attribute register, the AM code that the master places on the VMEbus can be programmed under software control.

The VMEchip2 includes a software-controlled VMEbus access timer, and it starts ticking when the chip is requested to do a VMEbus data transfer or an interrupt acknowledge cycle. The timer stops ticking once the chip has started the data transfer on the VMEbus. If the data transfer does not begin before the timer times out, the timer drives the local peripheral bus error signal, and sets the appropriate status bit in the Local Control and Status Register (LCSR). Using control bits in the LCSR, the timer can be disabled, or it can be enabled to drive the local peripheral bus error signal after 64 µsecs, 1 msec, or 32 msecs.

The VMEchip2 includes a software-controlled VMEbus write post timer, and it starts ticking when a data transfer to the VMEbus is write posted. The timer stops ticking once the chip has started the data transfer on the VMEbus. If this does not happen before the timer times out, the chip aborts the write posted cycle and send an interrupt to the local peripheral bus interrupter. If the write post bus error interrupt is enabled in the local peripheral bus interrupter, the local interrupt handler is interrupted to indicate a write post timeout has occurred. The write post timer has the same timing as the VMEbus access timer.

Local Peripheral Bus to VMEbus Requester

The requester provides all the signals necessary to allow the local peripheral bus to VMEbus master to request and be granted use of the VMEbus. The chip connects to all signals that a VMEbus requester is required to drive and monitor.

Requiring no external jumpers, the chip provides the means for software to program the requester to request the bus on any one of the four bus request levels, automatically establishing the bus grant daisy-chains for the three inactive levels.

The requester requests the bus if any of the following conditions occur:

1. The local peripheral bus master initiates either a data transfer cycle or an interrupt acknowledge cycle to the VMEbus.
2. The chip is requested to acquire control of the VMEbus as signaled by the DWB input signal pin.
3. The chip is requested to acquire control of the VMEbus as signaled by the DWB control bit in the LCSR.

The local peripheral bus to VMEbus requester in the VMEchip2 implements a FAIR mode. By setting the FAIR bit, the requester refrains from requesting the VMEbus until it detects its assigned request line in its negated state.

The local peripheral bus to VMEbus requester attempts to release the VMEbus when the requested data transfer operation is complete, the DWB pin is negated, the DWB bit in the LCSR is negated and the bus is not being held by a lock cycle. The requester releases the bus as follows:

1. When the chip is configured in the release-when-done (RWD) mode, the requester releases the bus when the above conditions are satisfied.
2. When the chip is configured in the release-on-request (ROR) mode, the requester releases the bus when the above conditions are satisfied and there is a bus request pending on one of the VMEbus request lines.

To minimize the timing overhead of the arbitration process, the local peripheral bus to VMEbus requester in the VMEchip2 executes an early release of the VMEbus. If it is about to release the bus and it is executing a VMEbus cycle, the requester releases BBSY before its associated master completes the cycle. This allows the arbiter to arbitrate any pending requests, and grant the bus to the next requester, at the same time that the active master completes its cycle.

VMEbus to Local Peripheral Bus Interface

The VMEbus to local peripheral bus interface allows an offboard VMEbus master access to onboard resources. The VMEbus to local peripheral bus interface includes the VMEbus slave, write post buffer, and local peripheral bus master. Adhering to the IEEE 1014-87 VMEbus Standard, the slave can withstand address-only cycles, as well as address pipelining, and respond to unaligned transfers. Using programmable map decoders, it can be configured to provide the following VMEbus capabilities:

Addressing capabilities:	A24, A32
Data transfer capabilities:	D08(EO), D16, D32, D8/BLT, D16/BLT, D32/BLT, D64/BLT (BLT = block transfer)

The slave can be programmed to perform write posting operations. When in this mode, the chip latches incoming data and addressing information into a staging FIFO and then acknowledges the VMEbus write transfer by asserting DTACK. The chip then requests control of the local peripheral bus and independently accesses the local resource after it has been granted the local peripheral bus. The write-posting pipeline is two deep in the non-block transfer mode and 16 deep in the block transfer mode.

To significantly improve the access time of the slave when it responds to a VMEbus block read cycle, the VMEchip2 contains a 16 four-byte deep read-ahead pipeline. When responding to a block read cycle, the chip performs block read cycles on the local peripheral bus to keep the FIFO buffer full. Data for subsequent transfers is then retrieved from the onchip buffer, significantly improving the response time of the slave in the block transfer mode.

The VMEchip2 includes an onchip map decoder that allows software to configure the global addressing range of onboard resources. The decoder allows the local peripheral address range to be partitioned into two separate banks, each with its own start and end address (in increments of 64KB), as well as set each bank's address modifier codes and write post enable and snoop enable.

Each map decoder includes an alternate address register and an alternate address select register. These registers allow any or all of the upper 16 VMEbus address lines to be replaced by signals from the alternate address register. This allows the address of local resources to be different from their VMEbus address.

The alternate address register also provides the upper eight bits of the local address when the VMEbus slave cycle is A24.

The local peripheral bus master requests the local peripheral bus and executes cycles as required. To reduce local peripheral bus loading and improve performance it always attempts to transfer data using a burst transfer as defined by the MC68040.

Local Peripheral Bus to VMEbus DMA Controller

The DMA Controller operates in conjunction with the local peripheral bus master, the VMEbus master, and a 16 four-byte FIFO buffer. The DMA controller has a 32-bit local address counter, 32-bit table address counter, a 32-bit VMEbus address counter, a 32-bit byte counter, and control and status registers. The Local Control and Status Register (LCSR) provides software with the ability to control the operational modes of the DMAC. Software can program the DMAC to transfer up to 4GB of data in the course of a single DMA operation. The DMAC supports transfers from any local peripheral bus address to any VMEbus address. The transfers may be from one byte to 4GB in length.

To optimize local peripheral bus use, the DMAC automatically adjusts the size of individual data transfers until 32-bit transfers can be executed. Based on the address of the first byte, the DMAC transfers a single-byte, a double-byte, or a mixture of both, and then continues to execute quad-byte block transfer cycles. When the DMAC is set for 64-bit transfers, the octal-byte transfers takes place. Based on the address of the last byte, the DMAC transfers a single-byte, a double-byte, or a mixture of both to end the transfer.

Using control register bits in the LCSR, the DMAC can be configured to provide the following VMEbus capabilities:

Addressing capabilities:	A16, A24, A32
Data transfer capabilities:	D16, D32, D16/BLT, D32/BLT, D64/BLT (BLT = block transfer)

Using the DMA AM control register, the address modifier code that the VMEbus DMA Controller places on the VMEbus can be programmed under software control. In addition, the DMAC can be programmed to execute block-transfer cycles over the VMEbus.

Complying with the VMEbus specification, the DMAC automatically terminates block-transfer cycles whenever a 256-Byte (D32/BLT) or 512-KByte (D64/BLT) boundary is crossed. It does so by momentarily releasing AS and then, in accordance with its bus release/bus request configuration, initiating a new block-transfer cycle.

To optimize VMEbus use, the DMAC automatically adjusts the size of individual data transfers until 64-bit transfers (D64/BLT mode), 32-bit transfers (D32 mode) or 16-bit transfers (D16 mode) can be executed. Based on the address of the first byte, the DMAC transfers single-byte, double-byte, or a mixture of both, and then continues to execute transfer cycles based on the programmed data width. Based on the address of the last byte, the DMAC transfers single-byte, double-byte, or a mixture of both to end the transfer.

To optimize local peripheral bus use when the VMEbus is operating in the D16 mode, the data FIFO converts D16 VMEbus transfers to D32 local peripheral bus transfers. The FIFO also aligns data if the source and destination addresses are not aligned so the local peripheral bus and VMEbus can operate at their maximum data transfer sizes.

To allow other boards access to the VMEbus, the DMAC has bus tenure timers to limit the time the DMAC spends on the VMEbus and to ensure a minimum time off the VMEbus. Since the local peripheral bus is generally faster than the VMEbus, other local peripheral bus masters may use the local peripheral bus while the DMAC is waiting for the VMEbus.

The DMAC also supports command chaining through the use of a singly-linked list built in local memory. Each entry in the list includes a VMEbus address, a local peripheral bus address, a byte count, a control word, and a pointer to the next entry. When the command chaining mode is enabled, the DMAC reads and executes commands from the list in local memory until all commands are executed.

The DMAC can be programmed to send an interrupt request to the local peripheral bus interrupter when any specific table entry has completed. In addition the DMAC always sends an interrupt request at the normal completion of a request or when an error is detected. If the DMAC interrupt is enabled in the DMAC, the local peripheral bus is interrupted.

To allow increased flexibility in managing the bus tenure to optimize bus usage as required by the system configuration, the chip contains control bits that allow the DMAC time on and off the bus to be programmed. Using these control bits, software can instruct the DMA Controller to acquire the bus, maintain mastership for a specific amount of time, and then, after relinquishing it, refrain from requesting it for another specific amount of time.

The DMAC also supports command chaining through the use of a singly-linked list built in local memory. Each entry in the list includes a VMEbus address, a local peripheral bus address, a byte count, a control word, and a pointer to the next entry. When the command chaining mode is enabled, the DMAC reads and executes commands from the list in local memory until all commands are executed.

The DMAC can be programmed to send an interrupt request to the local peripheral bus interrupter when any specific table entry has completed. In addition the DMAC always sends an interrupt request at the normal completion of a request or when an error is detected. If the DMAC interrupt is enabled in the DMAC, the local peripheral bus is interrupted.

To allow increased flexibility in managing the bus tenure to optimize bus usage as required by the system configuration, the chip contains control bits that allow the DMAC time on and off the bus to be programmed. Using these control bits, software can instruct the DMA Controller to acquire the bus, maintain mastership for a specific amount of time, and then, after relinquishing it, refrain from requesting it for another specific amount of time.

DMAC VMEbus Requester

The chip contains an independent VMEbus requester associated with the DMA Controller. This allows flexibility in instituting different bus tenure policies for the single-transfer oriented master, and the block-transfer oriented DMA controller. The DMAC requester provides all the signals necessary to allow the on-chip DMA Controller to request and be granted use of the VMEbus.

Requiring no external jumpers, the chip provides the means for software to program the DMAC requester to request the bus on any one of the four bus request levels, automatically establishing the bus grant daisy-chains for the three inactive levels.

The DMAC requester requests the bus as required to transfer data to or from the FIFO buffer.

The requester implements a FAIR mode. By setting the FAIR bit, the requester refrains from requesting the bus until it detects its assigned request line in its negated state.

The requester releases the bus when requested to by the DMA controller. The DMAC always releases the VMEbus when the FIFO is full (VMEbus to local peripheral bus) or empty (local peripheral bus to VMEbus). The DMAC can also be programmed to release the VMEbus when another VMEbus master requests the bus, when the time on timer has expired, or when the time on timer has expired and another VMEbus master is requesting the bus. To minimize the timing overhead of the arbitration process, the DMAC requester executes an early release of the bus. If it is about to release the bus and it is executing a VMEbus cycle, the requester releases BBSY before its associated VMEbus master completes the cycle. This allows the arbiter to arbitrate any pending requests, and grant the bus to the next requester, at the same time that the DMAC completes its cycle.

Tick and Watchdog Timers

The VMEchip2 has two 32-bit tick timers and a watchdog timer. The tick timers run on a 1 MHz clock which is derived from the processor clock by the prescaler.

Prescaler

The prescaler is used to derive the various clocks required by the tick timers, VME access timers, reset timer, bus arbitration timer, local peripheral bus timer, and VMEbus timer. The prescaler divides the local peripheral bus clock to produce the constant-frequency clocks required. Software is required to load the appropriate constant, depending upon the local peripheral bus clock, following reset to ensure proper operation of the prescaler.

Tick Timer

The VMEchip2 includes two general purpose tick timers. These timers can be used to generate interrupts at various rates or the counters can be read at various times for interval timing. The timers have a resolution of 1 μs and when free running, they roll over every 71.6 minutes.

Each tick timer has a 32-bit counter, a 32-bit compare register, a 4-bit overflow register, an enable bit, an overflow clear bit, and a clear-on-compare enable bit. The counter is readable and writable at any time and when enabled in the free run mode, it increments every 1 μsec. When the counter is enabled in the clear-on-compare mode, it increments every 1 μsec until the counter value matches the value in the compare register. When a match occurs, the counter is cleared. When a match occurs, in either mode, an interrupt is sent to the local peripheral bus interrupter and the overflow counter is incremented. An interrupt to the local peripheral bus is only generated if the tick timer interrupt is enabled by the local peripheral bus interrupter. The overflow counter can be cleared by writing a one to the overflow clear bit.

Tick timer one or two can be programmed to generate a pulse on the VMEbus IRQ1 interrupt line at the tick timer period. This provides a broadcast interrupt function which allows several VME boards to receive an interrupt at the same time. In certain applications, this interrupt can be used to synchronize multiple processors. This interrupt is not acknowledged on the VMEbus. This mode is intended for specific applications and is not defined in the VMEbus specification.

Watchdog Timer

The watchdog timer has a 4-bit counter, four clock select bits, an enable bit, a local reset enable bit, a SYSREST enable bit, a board fail enable bit, counter reset bit, WDTO status bit, and WDTO status reset bit.

When enabled, the counter increments at a rate determined by the clock select bits. If the counter is not reset by software, the counter reaches its terminal count. When this occurs, the WDTO status bit is set; and if the local or SYSRESET function is enabled, the selected reset is generated; if the board fail function is enabled, the board fail signal is generated.

VMEbus Interrupter

The interrupter provides all the signals necessary to allow software to request interrupt service from a VMEbus interrupt handler. The chip connects to all signals that a VMEbus interrupter is required to drive and monitor.

Requiring no external jumpers, the chip provides the means for software to program the interrupter to request an interrupt on any one of the seven interrupt request lines. In addition, the chip controls the propagation of the acknowledge on the IACK daisy-chain.

The interrupter operates in the release-on-acknowledge (ROAK) mode. An 8-bit control register provides software with the means to dynamically program the status/ID information. Upon reset, this register is initialized to a status/ID of $0F (the uninitialized vector in the 68K-based environment).

The VMEbus interrupter has an additional feature not defined in the VMEbus specification. The VMEchip2 supports a broadcast mode on the IRQ1 signal line. When this feature is used, the normal IRQ1 interrupt to the local peripheral bus interrupter should be disabled and the edge-sensitive IRQ1 interrupt to the local peripheral bus interrupter should be enabled. All boards in the system which are not participating in the broadcast interrupt function should not drive or respond to any signals on the IRQ1 signal line.

There are two ways to broadcast an IRQ1 interrupt. The VMEbus interrupter in the VMEchip2 may be programmed to generate a level one interrupt. This interrupt must be cleared using the interrupt clear bit in the control register because the interrupt is never acknowledged on the VMEbus. The VMEchip2 allows the output of one of the tick timers to be connected to the IRQ1 interrupt signal line on the VMEbus. When this function is enabled, a pulse appears on the IRQ1 signal line at the programmed interrupt rate of the tick timer.

VMEbus System Controller

With the exception of the optional SERCLK Driver and the Power Monitor, the chip includes all the functions that a VMEbus System Controller must provide. The System Controller is enabled/disabled with the aid of an external jumper (the only jumper required in a VMEchip2 based VMEbus interface).

Arbiter

The arbitration algorithm used by the chip arbiter is selected by software. All three arbitration modes defined in the VMEbus specification are supported: Priority (PRI), Round-Robin-Select (RRS), as well as Single (SGL). When operating in the PRI mode, the arbiter asserts the BCLR line whenever it detects a request for the bus whose level is higher than the one being serviced.

The chip includes an arbitration timer, preventing a bus lock-up when no requester assumes control of the bus after the arbiter has issued a grant. Using a control bit, this timer can be enabled or disabled. When enabled, it assumes control of the bus by driving the BBSY signal after 256 μsecs, releasing it after satisfying the requirements of the VMEbus specification, and then re-arbitrating any pending bus requests.

IACK Daisy-Chain Driver

Complying with the latest revision of the VMEbus specification, the System Controller includes an IACK Daisy-Chain Driver, ensuring that the timing requirements of the IACK daisy-chain are satisfied.

Bus Timer

The Bus Timer is enabled/disabled by software to terminate a VMEbus cycle by asserting BERR if any of the VMEbus data strobes is maintained in its asserted state for longer than the programmed timeout period. The timeout period can be set to 8, 64, or 256 μsecs. The bus timer terminates an unresponsive VMEbus cycle only if both it and the system controller are enabled.

In addition to the VMEbus timer, the chip contains a local peripheral bus timer. This timer asserts the local TEA when the local peripheral bus cycle maintained in its asserted state for longer than the programmed timeout period. This timer can be enabled or disabled under software control. The timeout period can be programmed for 8, 64, or 256 µsecs.

Reset Driver

The chip includes both a global and a local reset driver. When the chip operates as the VMEbus system controller, the reset driver provides a global system reset by asserting the VMEbus signal SYSRESET. The SYSRESET signal may be generated by the reset switch, a power up reset, a watch dog timeout, or by a control bit in the LCSR. SYSRESET remains asserted for at least 200 msecs, as required by the VMEbus specification.

Similarly, the chip provides an input signal and a control bit to initiate a local reset operation. By setting a control bit, software can maintain a board in a reset state, disabling a faulty board from participating in normal system operation. The local reset driver is enabled even when the chip is not the system controller. A local reset may be generated by the reset switch, a power up reset, a watch dog timeout, a VMEbus SYSRESET, or a control bit in the GCSR. This local reset signal goes to the BusSwitch and the BusSwitch combines the VMEchip2 reset, the reset switch, and the power up reset to generate a local board reset.

Local Peripheral Bus Interrupter and Interrupt Handler

There are 31 interrupt sources in the VMEchip2: VMEbus ACFAIL, ABORT switch, VMEbus SYSFAIL, write post bus error, external input, VMEbus IRQ1 edge-sensitive, VMEchip VMEbus interrupter acknowledge, tick timer 2-1, DMAC done, GCSR SIG3-0, GCSR location monitor 1-0, software interrupts 7-0, and VMEbus IRQ7-1. Each of the 31 interrupts can be enabled to generate a local peripheral bus interrupt at any level. For example, VMEbus IRQ5 can be programmed to generate a level 2 local peripheral bus interrupt.

The VMEbus AC fail interrupter is an edge-sensitive interrupter connected to the VMEbus ACFAIL signal line. This interrupter is filtered to remove the ACFAIL glitch which is related to the BBUSY glitch.

The ABORT switch interrupter is an edge-sensitive interrupter connected to the ABORT switch. This interrupter is filtered to remove switch bounce.

The SYS fail interrupter is an edge-sensitive interrupter connected to the VMEbus SYSFAIL signal line.

The write post bus error interrupter is an edge-sensitive interrupter connected to the local peripheral bus to VMEbus write post bus error signal line.

The external interrupter is an edge-sensitive interrupter connected to a signal pin on the VMEchip2.

The VMEbus IRQ1 edge-sensitive interrupter is an edge-sensitive interrupter connected to the VMEbus IRQ1 signal line. This interrupter is used when one of the tick timers is connected to the IRQ1 signal line. When this interrupt is acknowledged, the vector is provided by the VMEchip2 and a VMEbus interrupt acknowledge is not generated. When this interrupt is enabled, the VMEbus IRQ1 level-sensitive interrupter should be disabled.

The VMEchip2 VMEbus interrupter acknowledge interrupter is an edge-sensitive interrupter connected to the acknowledge output of the VMEbus interrupter. An interrupt is generated when an interrupt on the VMEbus from VMEchip2 is acknowledged by a VMEbus interrupt handler.

The tick timer interrupters are edge-sensitive interrupters connected to the output of the tick timers.

The DMAC interrupter is an edge-sensitive interrupter connected to the DMAC.

The GCSR SIG3-0 interrupters are edge-sensitive interrupters connected to the output of the signal bits in the GCSR.

The location monitor interrupters are edge-sensitive interrupters connected to the location monitor bits in the GCSR.

The software 7-0 interrupters can be set by software to generate interrupts.

The VMEbus IRQ7-1 interrupters are level-sensitive interrupters connected to the VMEbus IRQ7-1 signal lines.

The interrupt handler provides all logic necessary to identify and handle all local interrupts as well as VMEbus interrupts. When a local interrupt is acknowledged, a unique vector is provided by the chip. Edge-sensitive interrupters are not cleared during the interrupt acknowledge cycle and must by reset by software as required. If the interrupt source is the VMEbus, the interrupt handler instructs the VMEbus master to execute a VMEbus IACK cycle to obtain the vector from the VMEbus interrupter. The chip connects to all signals that a VMEbus handler is required to drive and monitor. On the local peripheral bus, the interrupt handler is designed to comply with the interrupt handling signaling protocol of the MC68040 microprocessor.

Global Control and Status Registers

The VMEchip2 includes a set of registers that are accessible from both the VMEbus and the local peripheral bus. These registers are provided to aid in interprocessor communications over the VMEbus. These registers are fully described in a later section.

VMEboard Functions

The VMEchip2 also includes several functions that are generally used on VMEbus boards. The VMEchip2 provides eight general purpose input signal pins and four general purpose I/O pins.

LCSR Programming Model

This section defines the programming model for the Local Control and Status Registers (LCSR) in the VMEchip2. The local peripheral bus map decoder for the LCSR is included in the VMEchip2. The base address of the LCSR is $FFF40000 and the registers are 32-bits wide. Byte, two-byte and four-byte read operations are permitted: however, byte and two-byte write operations are not permitted. Byte and two-byte write operations return a TEA signal to the local peripheral bus. Read-modify-write operations should be used to modify a byte or a two-byte of a register.

Each register definition includes a table with 5 lines. Line 1 is the base address of the register and the number of bits defined in the table. Line 2 shows the bits defined by this table. Line 3 defines the name of the register or the name of the bits in the register. Line 4 defines the operations possible on the register bits as follows:

1. R - This bit is a read-only status bit.
2. R/W - This bit is readable and writable.
3. W/AC - This bit can be set and it is automatically cleared. This bit can also be read.
4. C - Writing a one to this bit clears this bit or another bit. This bit reads zero.
5. S - Writing a one to this bit sets this bit or another bit. This bit reads zero.

Line 5 defines the state of the bit following a reset as defined below.

1. P - The bit is affected by power-up reset.
2. S - The bit is affected by SYSRESET.
3. L - The bit is affected by local reset.
4. X - The bit is not affected by reset.

A summary of the LCSR is shown Table 2-1.

Table 2-1. VMEchip2 Memory Map - LCSR Summary (Sheet 1 of 3)

VMEchip2 LCSR Base Address = $FFF40000

OFFSET:

	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
00	VMEbus SLAVE ENDING ADDRESS 1																VMEbus SLAVE STARTING ADDRESS 1															
04	VMEbus SLAVE ENDING ADDRESS 2																VMEbus SLAVE STARTING ADDRESS 2															
08	VMEbus SLAVE ADDRESS TRANSLATION ADDRESS 1																VMEbus SLAVE ADDRESS TRANSLATION SELECT 1															
0C	VMEbus SLAVE ADDRESS TRANSLATION ADDRESS 2																VMEbus SLAVE ADDRESS TRANSLATION SELECT 2															
10						(VB) SNP 2		(VB) WP 2	(VB) SUP 2	(VB) USR 2	(VB) A32 2	(VB) A24 2	(VB) D64 2	(VB) BLK 2	(VB) PGM 2	(VB) DAT 2						(VB) SNP 1		(VB) WP 1	(VB) SUP 1	(VB) USR 1	(VB) A32 1	(VB) A24 1	(VB) D64 1	(VB) BLK 1	(VB) PGM 1	(VB) DAT 1
14	LOCAL BUS SLAVE ENDING ADDRESS 1																LOCAL BUS SLAVE STARTING ADDRESS 1															
18	LOCAL BUS SLAVE ENDING ADDRESS 2																LOCAL BUS SLAVE STARTING ADDRESS 2															
1C	LOCAL BUS SLAVE ENDING ADDRESS 3																LOCAL BUS SLAVE STARTING ADDRESS 3															
20	LOCAL BUS SLAVE ENDING ADDRESS 4																LOCAL BUS SLAVE STARTING ADDRESS 4															
24	LOCAL BUS SLAVE ADDRESS TRANSLATION ADDRESS 4																LOCAL BUS SLAVE ADDRESS TRANSLATION SELECT 4															
28	(LB) D16 EN	(LB) WP EN	(LB) AM 4						(LB) D16 EN	(LB) WP EN	(LB) AM 3						(LB) D16 EN	(LB) WP EN	(LB) AM 2						(LB) D16 EN	(LB) WP EN	(LB) AM 1					
2C	(VB) GCSR GROUP ADDRESS								(VB) GCSR BOARD ADDRESS				LB EN4	LB EN3	LB EN2	LB EN1	LB I2 EN	LB I2 WP	LB I2 SU	LB I2 PD	LB I1 EN	LB I1 D16	LB I1 WP	LB I1 SU	ROM SIZE (XX)		ROM BANK B SPEED (XX)			ROM BANK A SPEED (XX)		
	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

LB = Local Bus

(LB) = Local Bus Slave

LV = Local Bus to VMEbus

VB = VMEbus

(VB) = VMEbus Slave

(XX) = Not Used on the MVME197 Series

Table 2-1. VMEchip2 Memory Map - LCSR Summary (Continued) (Sheet 2 of 3)

VMEchip2 LCSR Base Address = $FFF40000

OFFSET:

	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
30												ROMO (XX)	DMAC TB SNP MODE (XX)		SPRAM SPEED (XX)		LV ROBN MODE	LV DHB	LV DWB		LV FAIR	LV RWD	LV REQUEST LEVEL		DMAC HALT	DMAC EN	DMAC TBL	DMAC FAIR	DMAC RELM		DMAC REQUEST LEVEL	
34																	DMAC INTE	DMAC LB SNP MODE			DMAC VME INC	DMAC LB INC	DMAC TVME	DMAC D16	DMAC BLK TR	DMAC BLK TR	DMAC AM	DMAC AM	DMAC AM	DMAC AM	DMAC AM	DMAC AM
38	DMAC LOCAL BUS ADDRESS COUNTER																															
3C	DMAC VMEbus ADDRESS COUNTER																															
40	DMAC BYTE COUNTER																															
44	DMAC TABLE ADDRESS COUNTER																															
48		INTERRUPT 1 SIGNAL		IRQ CLR	IRQ STAT	VMEbus INTERRUPT LEVEL			VMEbus INTERRUPT VECTOR								DMAC INTERRUPT COUNTER				MPU CLR	MPU LB EN	MPU LB PERR	MPU LB OB	MPU LB TO	DMAC LB EN	DMAC LB PERR	DMAC LB OB	DMAC LB TO	DMAC LB ERR	DMAC VME ERR	DMAC DONE
4C								VB TO	DMAC TIME OFF			DMAC TIME ON			VMEbus GLOBAL TIMEOUT		VMEbus ACCESS TIMEOUT		LOCAL BUS TIMEOUT		WATCHDOG TIMEOUT				PRESCALER ADJUST							
50	TICK TIMER 1 COMPARE																															
54	TICK TIMER 1 COUNTER																															
58	TICK TIMER 2 COMPARE																															
5C	TICK TIMER 2 COUNTER																															
60		SCON	SYS FAIL	BRD FAIL STAT	PURS STAT	CLR PURS STAT	BRD FAIL OUT	RST SW EN	SYS RST	WD CLR STAT	WD CLR CNT	WD TO STAT	WD BDF EN	WD SYS LRST	WD RST EN	WD EN	OVERFLOW COUNTER 2					CLR OVF 2	COC 2	CNT EN 2	OVERFLOW COUNTER 1					CLR OVF 1	COC 1	CNT EN 1
64	PRESCALER COUNTER																															
	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

LB	= Local Bus		VB	= VMEbus
(LB)	= Local Bus Slave		(VB)	= VMEbus Slave
LV	= Local Bus to VMEbus		(XX)	= Not Used on the MVME197 Series

Table 2-1. VMEchip2 Memory Map - LCSR Summary (Continued) (Sheet 3 of 3)

VMEchip2 LCSR Base Address = $FFF40000
OFFSET:

	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
68	AC FAIL IRQ	AB SW IRQ	SYS FAIL IRQ	MWP ERR IRQ	PE IRQ	IRQ1 EDGE IRQ	TIC TIM2 IRQ	TIC TIM1 IRQ	VME IACK IRQ	DMAC IRQ	GCSR SIG3 IRQ	GCSR SIG2 IRQ	GCSR SIG1 IRQ	GCSR SIG0 IRQ	GCSR LM1 IRQ	GCSR LM0 IRQ	LB SW7 IRQ	LB SW6 IRQ	LB SW5 IRQ	LB SW4 IRQ	LB SW3 IRQ	LB SW2 IRQ	LB SW1 IRQ	LB SW0 IRQ	SPARE	VME IRQ7 IRQ	VME IRQ6 IRQ	VME IRQ5 IRQ	VME IRQ4 IRQ	VME IRQ3 IRQ	VME IRQ2 IRQ	VME IRQ1 IRQ
6C	EN IRQ 31	EN IRQ 30	EN IRQ 29	EN IRQ 28	EN IRQ 27	EN IRQ 26	EN IRQ 25	EN IRQ 24	EN IRQ 23	EN IRQ 22	EN IRQ 21	EN IRQ 20	EN IRQ 19	EN IRQ 18	EN IRQ 17	EN IRQ 16	EN IRQ 15	EN IRQ 14	EN IRQ 13	EN IRQ 12	EN IRQ 11	EN IRQ 10	EN IRQ 9	EN IRQ 8	EN IRQ 7	EN IRQ 6	EN IRQ 5	EN IRQ 4	EN IRQ 3	EN IRQ 2	EN IRQ 1	EN IRQ 0
70																	SET IRQ 15	SET IRQ 14	SET IRQ 13	SET IRQ 12	SET IRQ 11	SET IRQ 10	SET IRQ 9	SET IRQ 8								
74	CLR IRQ 31	CLR IRQ 30	CLR IRQ 29	CLR IRQ 28	CLR IRQ 27	CLR IRQ 26	CLR IRQ 25	CLR IRQ 24	CLR IRQ 23	CLR IRQ 22	CLR IRQ 21	CLR IRQ 20	CLR IRQ 19	CLR IRQ 18	CLR IRQ 17	CLR IRQ 16	CLR IRQ 15	CLR IRQ 14	CLR IRQ 13	CLR IRQ 12	CLR IRQ 11	CLR IRQ 10	CLR IRQ 9	CLR IRQ 8								
78		ACFAIL IRQ LEVEL				ABORT IRQ LEVEL				SYSFAIL IRQ LEVEL				MASTER WRITE POST ERROR IRQ LEVEL				PARITY ERROR IRQ LEVEL				IRQ1 EDGE-SENSITIVE IRQ LEVEL				TICK TIMER 2 IRQ LEVEL				TICK TIMER 1 IRQ LEVEL		
7C		VMEbus ACKNOWLEDGE IRQ LEVEL				DMAC IRQ LEVEL				GCSR SIG 3 IRQ LEVEL				GCSR SIG 2 IRQ LEVEL				GCSR SIG 1 IRQ LEVEL				GCSR SIG 0 IRQ LEVEL				GCSR LM 1 IRQ LEVEL				GCSR LM 0 IRQ LEVEL		
80		SW7 IRQ LEVEL				SW6 IRQ LEVEL				SW5 IRQ LEVEL				SW4 IRQ LEVEL				SW3 IRQ LEVEL				SW2 IRQ LEVEL				SW1 IRQ LEVEL				SW0 IRQ LEVEL		
84		SPARE IRQ LEVEL				VMEbus IRQ7 IRQ LEVEL				VMEbus IRQ6 IRQ LEVEL				VMEbus IRQ5 IRQ LEVEL				VMEbus IRQ4 IRQ LEVEL				VMEbus IRQ3 IRQ LEVEL				VMEbus IRQ2 IRQ LEVEL				VMEbus IRQ1 IRQ LEVEL		
88	VECTOR BASE REGISTER 0				VECTOR BASE REGISTER 1				MST IRQ EN	SYS FAIL LEVEL	AC FAIL LEVEL	ABORT LEVEL	GENERAL PURPOSE I/O ENABLE				GENERAL PURPOSE I/O OUTPUT				GENERAL PURPOSE I/O INPUT				GENERAL PURPOSE INPUT							
	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

LB	= Local Bus		VB	- VMEbus
(LB)	= Local Bus Slave		(VB)	= VMEbus Slave
LV	= Local Bus to VMEbus		(XX)	= Not Used on the MVME197 Series

Programming the VMEbus Slave Map Decoders

This section includes programming information for the VMEbus to local peripheral bus map decoders.

The VMEbus slave map decoders described in this section are disabled by SYSRESET or power-up reset. Caution must be used when enabling the map decoders or when modifying their registers after they are enabled. The safest time to enable or modify the map decoder registers is when the VMEchip2 is VMEbus master. The following procedure should be used to modify the map decoder registers. Set the DWB bit in the LCSR and then wait for the DHB bit in the LCSR to be set, indicating that VMEbus mastership has been acquired. The map decoder registers can then be modified and the VMEbus released by clearing the DWB bit in the LCSR. Because the VMEbus is held during this programming operation, the registers should be programmed quickly with interrupts disabled.

The VMEbus slave map decoders can be programmed, without obtaining VMEbus mastership, if they are disabled and the following procedure is followed. The address translation registers, starting and ending address registers should be programmed first, and then the map decoders should be enabled by programming the address modifier select registers.

The VMEbus to local peripheral bus interface allows offboard VMEbus masters access to local onboard resources. The address of the local resources as viewed from the VMEbus is controlled by the VMEbus slave map decoders, which are part of the VMEbus to local peripheral bus interface. Two VMEbus slave map decoders in the VMEchip2 allow two segments of the VMEbus to be mapped to the local peripheral bus. A segment may vary in size from 64KB to 4GB in increments of 64KB. Address translation is provided by the address translation registers which allow the upper 16 bits of the local peripheral bus address to be provided by the address translation address register rather than the upper 16 bits of the VMEbus.

Each VMEbus slave map decoder has an address translation address register, an address translation select register, a start address register, an end address register, an address modifier select register, and an attribute register. The addresses and bit definitions of these registers are shown in the following tables.

A VMEbus slave map decoder is programmed by loading the starting address of the segment into the starting address register and the ending address of the segment into the ending address register. If the VMEbus address modifier codes indicate an A24 VMEbus address cycle, then only the lower eight bits of the starting and ending address registers are used by the address comparator. The address modifier select register should be programmed for the required address modifier codes. A VMEbus slave map decoder is disabled when the address modifier select register is cleared.

The address translation registers allow local resources to have different VMEbus and local peripheral bus addresses. Only address bits A31 through A16 may be modified. The address translation registers also provide the upper eight local peripheral bus address lines when an A24 VMEbus cycle is used to accesses a local resource. The address translation register should be programmed with the translated address and the address translation select register should be programmed to enable the translated address. If address translation is not desired, then the address translation registers should be programmed to zero.

The address translation address register and the address translation select register operate in the following way. If a bit in the address translation select register is set, then the corresponding local peripheral bus address line is driven from the corresponding bit in the address translation address register. If the bit is cleared in the address translation select register, then the corresponding local peripheral bus address line is driven from the corresponding VMEbus address line. The most significant bit of the address translation select register corresponds to the most significant bit of address translation register and to A32 of the local peripheral bus and A32 of the VMEbus.

Write posting is enabled for the segment by setting the write post enable bit in the attribute register.

VMEbus Slave Ending Address Register 1

REG	VMEbus Slave Ending Address Register 1															
ADR/SIZ	$FFF40000 (16 bits of 32)															
BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	Slave Ending Address 1															
OPER	R/W															
RESET	0 PS															

This register is the ending address register for the first VMEbus to local peripheral bus map decoder.

VMEbus Slave Starting Address Register 1

REG	VMEbus Slave Starting Address Register 1															
ADR/SIZ	$FFF40000 (16 bits of 32)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	Slave Starting Address 1															
OPER	R/W															
RESET	0 PS															

This register is the starting address register for the first VMEbus to local peripheral bus map decoder.

VMEbus Slave Ending Address Register 2

REG	VMEbus Slave Ending Address Register 2															
ADR/SIZ	$FFF40004 (16 bits of 32)															
BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	Slave Ending Address 2															
OPER	R/W															
RESET	0 PS															

This register is the ending address register for the second VMEbus to local peripheral bus map decoder.

VMEbus Slave Starting Address Register 2

REG	VMEbus Slave Starting Address Register 2															
ADR/SIZ	$FFF40004 (16 bits of 32)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	Slave Starting Address 2															
OPER	R/W															
RESET	0 PS															

This register is the starting address register for the second VMEbus to local peripheral bus map decoder.

VMEbus Slave Address Translation Address Register 1

REG	VMEbus Slave Address Translation Address Register 1															
ADR/SIZ	$FFF40008 (16 bits of 32)															
BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	Slave Address Translation Address 1															
OPER	R/W															
RESET	0 PS															

This register is the address translation address register for the first VMEbus to local peripheral bus map decoder.

VMEbus Slave Address Translation Select Register 1

REG	VMEbus Slave Address Translation Select Register 1															
ADR/SIZ	$FFF40008 (16 bits of 32)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	Slave Address Translation Select 1															
OPER	R/W															
RESET	0 PS															

This register is the address translation select register for the first VMEbus to local peripheral bus map decoder.

VMEbus Slave Address Translation Address Register 2

REG	VMEbus Slave Address Translation Address Register 2															
ADR/SIZ	$FFF4000C (16 bits of 32)															
BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	Slave Address Translation Address 2															
OPER	R/W															
RESET	0 PS															

This register is the address translation address register for the second VMEbus to local peripheral bus map decoder.

VMEbus Slave Address Translation Select Register 2

REG	VMEbus Slave Address Translation Select Register 2															
ADR/SIZ	$FFF4000C (16 bits of 32)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	Slave Address Translation Select 2															
OPER	R/W															
RESET	0 PS															

This register is the address translation select register for the second VMEbus to local peripheral bus map decoder.

VMEbus Slave Write Post and Snoop Control Register 2

REG	VMEbus Slave Write Post and Snoop Control Register 2							
ADR/SIZ	$FFF40010 (8 bits [3 used] of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD						SNP2		WP2
OPER						R/W		R/W
RESET						0 PS		0 PS

This register is the address translation select register for the second VMEbus to local peripheral bus map decoder.

WP2 VMEbus Write Post 2. When this bit is high, write posting is enabled for the address range defined by the second VMEbus slave map decoder. When this bit is low, write posting is disabled for the address range defined by the second VMEbus slave map decoder.

SNP2 VMEbus Snoop 2. (This feature is not applicable to any of the MVME197 module series).

VMEbus Slave Address Modifier Select Register 2

REG	VMEbus Slave Address Modifier Select Register 2							
ADR/SIZ	$FFF40010 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD	SUP	USR	A32	A24	D64	BLK	PGM	DAT
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the address modifier select register for the second VMEbus to local peripheral bus map decoder. There are three groups of address modifier select bits: DAT, PGM, BLK and D64; A24 and A32; and USR and SUP. At least one bit must be set from each group to enable the map decoder.

DAT VMEbus Data 2. When this bit is high, the second map decoder responds to VMEbus data access cycles. When this bit is low, the second map decoder does not respond to VMEbus data access cycles.

PGM VMEbus Program 2. When this bit is high, the second map decoder responds to VMEbus program access cycles. When this bit is low, the second map decoder does not respond to VMEbus program access cycles.

BLK VMEbus Block 2. When this bit is high, the second map decoder responds to VMEbus block access cycles. When this bit is low, the second map decoder does not respond to VMEbus block access cycles.

D64 VMEbus Block D64 Access 2. When this bit is high, the second map decoder responds to VMEbus D64 block access cycles. When this bit is low, the second map decoder does not respond to VMEbus D64 block access cycles.

A24 VMEbus A24 Access 2. When this bit is high, the second map decoder responds to VMEbus A24 (standard) access cycles. When this bit is low, the second map decoder does not respond to VMEbus A24 access cycles.

A32 VMEbus A32 Access 2. When this bit is high, the second map decoder responds to VMEbus A32 (extended) access cycles. When this bit is low, the second map decoder does not respond to VMEbus A32 access cycles.

USR VMEbus User 2. When this bit is high, the second map decoder responds to VMEbus user (non-privileged) access cycles. When this bit is low, the second map decoder does not responded to VMEbus user access cycles.

SUP VMEbus Supervisory 2. When this bit is high, the second map decoder responds to VMEbus supervisory access cycles. When this bit is low, the second map decoder does not respond to VMEbus supervisory access cycles.

VMEbus Slave Write Post and Snoop Control Register 1

REG	VMEbus Slave Write Post and Snoop Control Register 1							
ADR/SIZ	$FFF40010 (8 bits [3 used] of 32)							
BIT	15	14	13	12	11	10	9	8
FIELD						SNP1		WP1
OPER						R/W		R/W
RESET						0 PS		0 PS

This register is the address translation select register for the first VMEbus to local peripheral bus map decoder.

WP1 VMEbus Write Post 1. When this bit is high, write posting is enabled for the address range defined by the first VMEbus slave map decoder. When this bit is low, write posting is disabled for the address range defined by the first VMEbus slave map decoder.

SNP1 VMEbus Snoop 1. (This feature is not applicable to any of the MVME197 module series).

VMEbus Slave Address Modifier Select Register 1

REG	VMEbus Slave Address Modifier Select Register 1							
ADR/SIZ	$FFF40010 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD	SUP	USR	A32	A24	D64	BLK	PGM	DAT
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the address modifier select register for the first VMEbus to local peripheral bus map decoder. There are three groups of address modifier select bits: DAT, PGM, BLK and D64; A24 and A32; and USR and SUP. At least one bit must be set from each group to enable the first map decoder.

DAT VMEbus Data 1. When this bit is high, the first map decoder responds to VMEbus data access cycles. When this bit is low, the first map decoder does not respond to VMEbus data access cycles.

PGM VMEbus Program 1. When this bit is high, the first map decoder responds to VMEbus program access cycles. When this bit is low, the first map decoder does not respond to VMEbus program access cycles.

BLK VMEbus Block 1. When this bit is high, the first map decoder responds to VMEbus block access cycles. When this bit is low, the first map decoder does not respond to VMEbus block access cycles.

D64 VMEbus Block D64 Access 1. When this bit is high, the first map decoder responds to VMEbus D64 block access cycles. When this bit is low, the first map decoder does not respond to VMEbus D64 block access cycles.

A24 VMEbus A24 Access 1. When this bit is high, the first map decoder responds to VMEbus A24 (standard) access cycles. When this bit is low, the first map decoder does not respond to VMEbus A24 access cycles.

A32 VMEbus A32 Access 1. When this bit is high, the first map decoder responds to VMEbus A32 (extended) access cycles. When this bit is low, the first map decoder does not respond to VMEbus A32 access cycles.

USR VMEbus User 1. When this bit is high, the first map decoder responds to VMEbus user (non-privileged) access cycles. When this bit is low, the first map decoder does not responded to VMEbus user access cycles.

SUP VMEbus Supervisory 1. When this bit is high, the first map decoder responds to VMEbus supervisory access cycles. When this bit is low, the first map decoder does not respond to VMEbus supervisory access cycles.

Programming the Local Peripheral Bus to VMEbus Map Decoders

This section includes programming information on the local peripheral bus to VMEbus map decoders and the GCSR base address registers.

The local peripheral bus to VMEbus interface allows onboard local peripheral bus masters access to offboard VMEbus resources. The address of the VMEbus resources as viewed from the local peripheral bus is controlled by the local peripheral bus slave map decoders, which are part of the local peripheral bus to VMEbus interface. Four of the six local peripheral bus to VMEbus map decoders are programmable, while the two I/O map decoders are fixed. The first I/O map decoder provides an A16/D16 or A16/D32 space at $FFFF0000 to $FFFFFFFF which is the VMEbus short I/O space. The second I/O map decoder provides an A24/D16 space at $F000000 to $F0FFFFFF and an A32/D16 space at $F1000000 to $FEFFFFFF.

A programmable segment may vary in size from 64KB to 4GB in increments of 64KB. Address translation for the fourth segment is provided by the address translation registers which allow the upper 16 bits of the VMEbus address to be provided by the address translation address register rather than the upper 16 bits of the local peripheral bus.

Each of the four programmable local peripheral bus map decoders has a starting address, an ending address, an address modifier register with attribute bits, and an enable bit. The fourth decoder also has address translation registers. The addresses and bit definitions for these registers are in the tables below.

A local peripheral bus slave map decoder is programmed by loading the starting address of the segment into the starting address register and the ending address of the segment into the ending address register. The address modifier code is programmed in to the address modifier register. Because the local peripheral bus to VMEbus interface does not support VMEbus block transfers, block transfer address modifier codes should not be programmed.

The address translation register allows a local peripheral bus master to view a portion of the VMEbus that may be hidden by onboard resources or an area of the VMEbus may be mapped to two local address. For example, some devices in the I/O map may support write posting while others do not. The VMEbus area in question may be mapped to two local peripheral bus addresses, one with write posting enabled and one with write posting disabled. The address translation registers allow local peripheral bus address bits A31 through A16 to be modified. The address translation register should be programmed with the translated address, and the address translation select register should be programmed to enable the translated address. If address translation is not desired, then the address translation registers should be programmed to zero.

The address translation address register and the address translation select register operate in the following way. If a bit in the address translation select register is set, then the corresponding VMEbus address line is driven from the corresponding bit in the address translation address register. If the bit is cleared in the address translation select register, then the corresponding VMEbus address line is driven from the corresponding local peripheral bus address line. The most significant bit of the address translation select register corresponds to the most significant bit of address translation address register and to A32 of the local peripheral bus and A32 of the VMEbus.

Write posting is enabled for the segment by setting the write post enable bit in the address modifier register. D16 transfers are forced by setting the D16 bit in the address modifier register. A segment is enabled by setting the enable bit. Segments should not be programmed to overlap.

The first I/O map decoder maps the local peripheral bus address range $FFFF0000 to $FFFFFFFF to the A16 (short I/O) map of the VMEbus. This segment may be enabled using the enable bit. Write posting may be enabled for this segment using the write post enable bit. The transfer size may be D16 or D32 as defined by the D16 bit in the control register.

The second I/O map decoder provides support for the other I/O map of the VMEbus. This decoder maps the local peripheral bus address range $F0000000 to $F0FFFFFF to the A24 map of the VMEbus and the address range $F1000000 to $FEFFFFFF to the A32 map of the VMEbus. The transfer size is always D16. This segment may be enabled using the enable bit. Write posting may be enabled using the write post enable bit.

The local peripheral bus map decoders should not be programmed such that more than one map decoder responds to the same local peripheral bus address or a map decoder conflicts with on board resources. However, the map decoders may be programmed to allow a VMEbus address to be accessed from more than one local peripheral bus address.

Local Peripheral Bus Slave Ending Address Register 1

REG	Local Peripheral Bus Slave Ending Address Register 1															
ADR/SIZ	$FFF40014 (16 bits of 32)															
BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	Slave Ending Address 1															
OPER	R/W															
RESET	0 PS															

This register is the ending address register for the first local peripheral bus to VMEbus map decoder.

Local Peripheral Bus Slave Starting Address Register 1

REG	Local Peripheral Bus Slave Starting Address Register 1															
ADR/SIZ	$FFF40014 (16 bits of 32)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	Slave Starting Address 1															
OPER	R/W															
RESET	0 PS															

This register is the starting address register for the first local peripheral bus to VMEbus map decoder.

Local Peripheral Bus Slave Ending Address Register 2

REG	Local Peripheral Bus Slave Ending Address Register 2															
ADR/SIZ	$FFF40018 (16 bits of 32)															
BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	Slave Ending Address 2															
OPER	R/W															
RESET	0 PS															

This register is the ending address register for the second local peripheral bus to VMEbus map decoder.

Local Peripheral Bus Slave Starting Address Register 2

REG	Local Peripheral Bus Slave Starting Address Register 2															
ADR/SIZ	$FFF40018 (16 bits of 32)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	Slave Starting Address 2															
OPER	R/W															
RESET	0 PS															

This register is the starting address register for the second local peripheral bus to VMEbus map decoder.

Local Peripheral Bus Slave Ending Address Register 3

REG	Local Peripheral Bus Slave Ending Address Register 3															
ADR/SIZ	$FFF4001C (16 bits of 32)															
BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	Slave Ending Address 3															
OPER	R/W															
RESET	0 PS															

This register is the ending address register for the third local peripheral bus to VMEbus map decoder.

Local Peripheral Bus Slave Starting Address Register 3

REG	Local Peripheral Bus Slave Starting Address Register 3															
ADR/SIZ	$FFF4001C (16 bits of 32)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	Slave Starting Address 3															
OPER	R/W															
RESET	0 PS															

This register is the starting address register for the third local peripheral bus to VMEbus map decoder.

Local Peripheral Bus Slave Ending Address Register 4

REG	Local Peripheral Bus Slave Ending Address Register 4															
ADR/SIZ	$FFF40020 (16 bits of 32)															
BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	Slave Ending Address 4															
OPER	R/W															
RESET	0 PS															

This register is the ending address register for the fourth local peripheral bus to VMEbus map decoder.

Local Peripheral Bus Slave Starting Address Register 4

REG	Local Peripheral Bus Slave Starting Address Register 4															
ADR/SIZ	$FFF40020 (16 bits of 32)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	Slave Starting Address 4															
OPER	R/W															
RESET	0 PS															

This register is the starting address register for the fourth local peripheral bus to VMEbus map decoder.

Local Peripheral Bus Slave Address Translation Address Register 4

REG	Local Peripheral Bus Slave Address Translation Address Register 4															
ADR/SIZ	$FFF40024 (16 bits of 32)															
BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	Slave Address Translation Address 4															
OPER	R/W															
RESET	0 PS															

This register is the address translation address register for the fourth local peripheral bus to VMEbus map decoder.

Local Peripheral Bus Slave Address Translation Select Register 4

REG	Local Peripheral Bus Slave Address Translation Select Register 4															
ADR/SIZ	$FFF40024 (16 bits of 32)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	Slave Address Translation Select 4															
OPER	R/W															
RESET	0 PS															

This register is the address translation select register for the fourth local peripheral bus to VMEbus map decoder.

Local Peripheral Bus Slave Attribute Register 4

REG	Local Peripheral Bus Slave Attribute Register 4							
ADR/SIZ	$FFF40028 (8 bits of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD	D16	WP	AM					
OPER	R/W	R/W	R/W					
RESET	0 PS	0 PS	0 PS					

This register is the attribute register for the fourth local peripheral bus to VMEbus map decoder.

AM Address Modifier 4. These bits define the VMEbus address modifier codes the VMEbus master uses for the segment defined by map decoder 4. Because the local peripheral bus to VMEbus interface does not support block transfers, the block transfer address modifier codes should not be used.

WP Write Posting 4. When this bit is high, write posting is enabled to the segment defined by map decoder 4. When this bit is low, write posting is disabled to the segment defined by map decoder 4.

D16 D16 Access 4. When this bit is high, D16 data transfers are performed to the segment defined by map decoder 4. When this bit is low, D32 data transfers are performed to the segment defined by map decoder 4.

Local Peripheral Bus Slave Attribute Register 3

REG	Local Peripheral Bus Slave Attribute Register 3							
ADR/SIZ	$FFF40028 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD	D16	WP	AM					
OPER	R/W	R/W	R/W					
RESET	0 PS	0 PS	0 PS					

This register is the attribute register for the third local peripheral bus to VMEbus map decoder.

AM Address Modifier 3. These bits define the VMEbus address modifier codes the VMEbus master uses for the segment defined by map decoder 3. Because the local peripheral bus to VMEbus interface does not support block transfers, the block transfer address modifier codes should not be used.

WP Write Posting 3. When this bit is high, write posting is enabled to the segment defined by map decoder 3. When this bit is low, write posting is disabled to the segment defined by map decoder 3.

D16 D16 Access 3. When this bit is high, D16 data transfers are performed to the segment defined by map decoder 3. When this bit is low, D32 data transfers are performed to the segment defined by map decoder 3.

Local Peripheral Bus Slave Attribute Register 2

REG	Local Peripheral Bus Slave Attribute Register 2							
ADR/SIZ	$FFF40028 (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
FIELD	D16	WP	AM					
OPER	R/W	R/W	R/W					
RESET	0 PS	0 PS	0 PS					

This register is the attribute register for the second local peripheral bus to VMEbus map decoder.

AM Address Modifier 2. These bits define the VMEbus address modifier codes the VMEbus master uses for the segment defined by map decoder 2. Since the local peripheral bus to VMEbus interface does not support block transfers, the block transfer address modifier codes should not be used.

WP Write Posting 2. When this bit is high, write posting is enabled to the segment defined by map decoder 2. When this bit is low, write posting is disabled to the segment defined by map decoder 2.

D16 D16 Access 2. When this bit is high, D16 data transfers are performed to the segment defined by map decoder 2. When this bit is low, D32 data transfers are performed to the segment defined by map decoder 2.

Local Peripheral Bus Slave Attribute Register 1

REG	Local Peripheral Bus Slave Attribute Register 1							
ADR/SIZ	$FFF40028 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD	D16	WP	AM					
OPER	R/W	R/W	R/W					
RESET	0 PS	0 PS	0 PS					

This register is the attribute register for the first local peripheral bus to VMEbus map decoder.

AM Address Modifier 1. These bits define the VMEbus address modifier codes the VMEbus master uses for the segment defined by map decoder 1. Because the local peripheral bus to VMEbus interface does not support block transfers, the block transfer address modifier codes should not be used.

WP Write Posting 1. When this bit is high, write posting is enabled to the segment defined by map decoder 1. When this bit is low, write posting is disabled to the segment defined by map decoder 1.

D16 D16 Access 1. When this bit is high, D16 data transfers are performed to the segment defined by map decoder 1. When this bit is low, D32 data transfers are performed to the segment defined by map decoder 1.

VMEbus Slave GCSR Group Address Register

REG	VMEbus Slave GCSR Group Address Register							
ADR/SIZ	$FFF4002C (8 bits of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD	GCSR Group							
OPER	R/W							
RESET	$00 PS							

This register defines the group address of the GCSR as viewed from the VMEbus. The GCSR address is defined by the group address and the board address. Once enabled, the GCSR register should not be reprogrammed unless the VMEchip2 is VMEbus master.

GCSR Group These bits define the group portion of the GCSR address. These bits are compared with VMEbus address lines A8 through A15. The recommended group address for the MVME197 is $CC.

VMEbus Slave GCSR Board Address Register

REG	VMEbus Slave GCSR Board Address Register							
ADR/SIZ	$FFF4002C (4 bits of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD	GCSR Board Address							
OPER	R/W							
RESET	$F PS							

This register defines the board address of the GCSR as viewed from the VMEbus. The GCSR address is defined by the group address and the board address. Once enabled, the GCSR register should not be reprogrammed unless the VMEchip2 is VMEbus master. The value $F in the GCSR board address register disables the map decoder. The map decoder is enabled when the board address is not $F.

GCSR Board These bits define the board number portion of the GCSR address. These bits are compared with VMEbus address lines A4 through A7. The GCSR is enabled by values $0 through $E. The address $XXFY in the VMEbus A16 space is reserved for the location monitors LM0 through LM3.

Note **XX is the group address and Y is the location monitor (1,LM0; 3,LM1; 5,LM2; 7,LM3).**

Local Peripheral Bus to VMEbus Enable Control Register

REG	Local Peripheral Bus to VMEbus Enable Control Register							
ADR/SIZ	$FFF4002C (4 bits of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD					EN4	EN3	EN2	EN1
OPER					R/W	R/W	R/W	R/W
RESET					0 PSL	0 PSL	0 PSL	0 PSL

This register is the map decoder enable register for the four programmable local peripheral bus to VMEbus map decoders.

EN1 Local Peripheral Bus Enable 1. When this bit is high, the first local peripheral bus to VMEbus map decoder is enabled. When this bit is low, the first local peripheral bus to VMEbus map decoder is disabled.

EN2 Local Peripheral Bus Enable 2. When this bit is high, the second local peripheral bus to VMEbus map decoder is enabled. When this bit is low, the second local peripheral bus to VMEbus map decoder is disabled.

EN3 Local Peripheral Bus Enable 3. When this bit is high, the third local peripheral bus to VMEbus map decoder is enabled. When this bit is low, the third local peripheral bus to VMEbus map decoder is disabled.

EN4 Local Peripheral Bus Enable 4. When this bit is high, the fourth local peripheral bus to VMEbus map decoder is enabled. When this bit is low, the fourth local peripheral bus to VMEbus map decoder is disabled.

Local Peripheral Bus to VMEbus I/O Control Register

REG	Local Peripheral Bus to VMEbus I/O Control Register							
ADR/SIZ	$FFF4002C (8 bits of 32)							
BIT	15	14	13	12	11	10	0	8
FIELD	I2EN	I2WP	I2SU	I2PD	I1EN	I1D16	I1WP	I1SU
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PS	0 PS	0 PS	0 PSL	0 PS	0 PS	0 PS

This register controls the VMEbus short I/O map and the F page ($F0000000 through $FEFFFFFF) I/O map.

I1SU I/O 1 Supervisor. When this bit is high, the VMEchip2 drives a supervisor address modifier code when the short I/O space is accessed. When this bit is low, the VMEchip2 drives a user address modifier code when the short I/O space is accessed.

I1WP I/O 1 Write Posting. When this bit is high, write posting is enabled to the VMEbus short I/O segment. When this bit is low, write posting is disabled to the VMEbus short I/O segment.

I1D16 I/O 1 D16 Access. When this bit is high, D16 data transfers are performed to the VMEbus short I/O segment. When this bit is low, D32 data transfers are performed to the VMEbus short I/O segment.

I1EN I/O 1 Enable. When this bit is high, the VMEbus short I/O map decoder is enabled. When this bit is low, the VMEbus short I/O map decoder is disabled.

I2PD I/O 2 Program. When this bit is high, the VMEchip2 drives a program address modifier code when the F page is accessed. When this bit is low, the VMEchip2 drives a data address modifier code when the F page is accessed.

I2SU I/O 2 Supervisor. When this bit is high, the VMEchip2 drives a supervisor address modifier code when the F page is accessed. When this bit is low, the VMEchip2 drives a user address modifier code when the F page is accessed.

I2WP I/O 2 Write Posting. When this bit is high, write posting is enabled to the local peripheral bus F page. When this bit is low, write posting is disabled to the local peripheral bus F page.

I2EN I/O 2 Enable. When this bit is high, the F page ($F0000000 through $FEFFFFFF) map decoder is enabled. The F0 page is defined as A24/D16 on the VMEbus while the F1-FE pages are defined as A32/D16. When this bit is low, the F page is disabled.

ROM Control Register

REG	ROM Control Register							
ADR/SIZ	$FFF4002C (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD	SIZE		BSPD			ASPD		
OPER	R/W		R/W			R/W		
RESET	0 PSL		0 PS			0 PS		

This register is the ROM control register. (This feature is not used on any of the MVME197 module series).

Programming the VMEchip2 DMA Controller

This section includes programming information on the DMA controller, VMEbus interrupter, MPU status register, and local peripheral bus to VMEbus requester register.

The VMEchip2 features a local peripheral bus - VMEbus DMA controller (DMAC). The DMAC has two modes of operation: command chaining, and direct. In the direct mode, the local peripheral bus address, the VMEbus address, the byte count, and the control register of the DMAC are programmed and the DMAC is enabled. The DMAC transfers data, as programmed, until the byte count is zero or an error is detected. When the DMAC stops, the status bits in the DMAC status register are set and an interrupt is sent to the local peripheral bus interrupter. If the DMAC interrupt is enabled in the local peripheral bus interrupter, the local peripheral bus is interrupted.

A maximum of 4GB of data may be transferred with one DMAC command. Larger transfers can be accomplished using the command chaining mode. In the command chaining mode, a singly-linked list of commands is built in local memory and the table address register in the DMAC is programmed with the starting address of the list of commands. The DMAC control register is programmed and the DMAC is enabled. The DMAC executes commands from the list until all commands are executed or an error is detected. When the

DMAC stops, the status bits are set in the DMAC status register and an interrupt is sent to the local peripheral bus interrupter. If the DMAC interrupt is enabled in the local peripheral bus interrupter, the local peripheral bus is interrupted. When the DMAC finishes processing a command in the list, and interrupts are enabled for that command, the DMAC sends an interrupt to the local peripheral bus interrupter. If the DMAC interrupt is enabled in the local peripheral bus interrupter, the local peripheral bus is interrupted.

The DMAC control is divided into two registers. The first register is only accessible by the processor. The second register can be loaded by the processor in the direct mode and by the DMAC in the command chaining mode.

Once the DMAC is enabled, the counter and control registers should not be modified by software. When the command chaining mode is used, the list of commands must be in local 32-bit memory and the entries must be four-byte aligned.

A DMAC command list includes one or more DMAC command packets. A DMAC command packet includes a control word that defines the VMEbus AM code, the VMEbus transfer size, the VMEbus transfer method, the DMA transfer direction, the VMEbus and local peripheral bus address counter operation, and the local peripheral bus snoop operation (note that the snoop operation feature is not used on the MVME197 module series). The format of the control word is the same as the lower 16 bits of the control register. The command packet also includes a local peripheral bus address, a VMEbus address, a byte count, and a pointer to the next command packet in the list. The end of a command is indicated by setting bit 0 or 1 of next command address. The command packet format is shown in Table 2-2.

Table 2-2. DMAC Command Table Format

Entry	Function	
0 (bits 0-15)	- - - -	Control Word
1 (bits 0-31)	Local Peripheral Bus Address	
2 (bits 0-31)	VMEbus Address	
3 (bits 0-31)	Byte Count	
4 (bits 0-31)	Address of Next Command Packet	

DMAC Registers

This section provides addresses and bit level descriptions of the DMAC counters, control registers, and status registers. Other control functions are also included in this section.

EPROM Decoder, SRAM and DMAC Control Register

REG	EPROM Decoder, SRAM and DMAC Control Register							
ADR/SIZ	$FFF40030 (8 bits [5 used] of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD				ROM0	TBLSC		SRAMS	
OPER				R/W	R/W		R/W	
RESET				1 PSL	0 PS		0 PS	

Refer to note below. This register controls the EPROM decoder, the snoop control bits used by the DMAC when it is accessing table entries, and the access time of the SRAM (Static RAM, also known as slow RAM). The time from SRAM CS* to data valid, for the SRAMs used with the VMEchip2, must be less than

(T * (bus clocks -1) - 35)

where **T** is the bus clock period, and **bus clocks** is the programmed number of local peripheral bus clocks. For example, if the bus clock is 33 MHz (30 nsecs), the number of bus clocks is 3, the access time of the SRAMs must be less than

(30 * (3-1) -35) = 25 nsecs.

Note **The MVME197 module series do not implement a SRAM and do not use the EPROM decoder. Instead, the MVME197 module series use the BusSwitch to generate Flash EPROM control.**

SRAMS Static RAM Speed. These bits define the number of bus clocks for a static RAM cycle.

SRAMS	Bus Clocks	Maximum SRAM Access Time at 25 MHz
0	6	165 nanosecond
1	5	125 nanosecond
2	4	85 nanosecond
3	4	45 nanosecond

TBLSC DMAC Table Snoop Mode. These bits control the snoop signal lines on the local peripheral bus when the DMAC is table walking. (This feature is not used on any of the MVME197 module series).

0 Snoop inhibited

1 Write - Sink data
Read - Supply dirty data and leave dirty

2 Write - Invalidate
Read - Supply dirty data and mark invalid

3 Snoop inhibited

ROM0 ROM 0. When this bit is set to 1, the EPROM decoder responds at $00000000 to $0003FFFF and $FF800000 to $FFBFFFFF. When this bit is set to 0, the EPROM decoder responds only at $FF800000 to $FFBFFFFF.

Note **ROM0 is set to 1 by power-up reset, SYSRESET, and local reset, causing ROM BANK A to provide reset vectors for the MPU. (This feature is not applicable to any of the MVME197 module series).**

Local Peripheral Bus to VMEbus Requester Control Register

REG	Local Peripheral Bus to VMEbus Requester Control Register							
ADR/SIZ	$FFF40030 (8 bits [7 used] of 32)							
BIT	15	14	13	12	11	10	0	8
FIELD	ROBN	DHB	DWB		LVFAIR	LVRWD	LVREQL	
OPER	R/W	R	R/W		R/W	R/W	R/W	
RESET	0 PS	0 PS	0 PSL		0 PS	0 PS	0 PS	

This register controls the VMEbus request level, the request mode, and release mode for the local peripheral bus to VMEbus interface.

LVREQL VMEbus Request Level. These bits define the VMEbus request level. The request is only changed when the VMEchip2 is bus master. The VMEchip2 always requests at the old level until it becomes bus master and the new level takes effect. If the VMEchip2 is bus master when the level is changed, the new level does not take effect until the bus has been released and re-requested at the old level. The requester always requests the VMEbus at level 3 the first time following a SYSRESET.

0 The request level is 0.

1 The request level is 1.

2 The request level is 2.

3 The request level is 3.

LVRWD Release-When-Done Mode. When this bit is high, the requester operates in the release-when-done mode. When this bit is low, the requester operates in the release-on-request mode.

LVFAIR Fair Mode. When this bit is high, the requester operates in the fair mode. When this bit is low, the requester does not operate in the fair mode. In the fair mode, the requester waits until the request signal line for the selected level is inactive before requesting the VMEbus.

DWB When this bit is high, the VMEchip2 requests the VMEbus and does not release it. When this bit is low, the VMEchip2 releases the VMEbus according to the release mode programmed in the LVRWD. When the VMEbus has been acquired, the DHB bit is set.

DHB When this bit is high, the VMEbus has been acquired in response to the DWB bit being set. When the DWB bit is cleared, this bit is cleared.

ROBN Round Robin Mode. When this bit is high, the VMEbus arbiter operates in the round robin mode. When this bit is low, the arbiter operates in the priority mode.

DMAC Control Register 1 (bits 0-7)

REG	DMAC Control Register 1							
ADR/SIZ	$FFF40030 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD	DHALT	DEN	DTBL	DFAIR	DRELM		DREQL	
OPER	S	S	R/W	R/W	R/W		R/W	
RESET	0 PS	0 PS	0 PS	0 PS	0 PS		0 PS	

This control register is loaded by the processor; it is not modified when the DMAC loads new values from the command packet.

DREQL DMAC Request Level. These bits define the VMEbus request level for the DMAC requester. The request is only changed when the VMEchip2 is bus master. The VMEchip2 always requests at the old level until it becomes bus master and the new level takes effect. If the VMEchip2 is bus master when the level is changed, the new level does not take effect until the bus has been released and re-requested at the old level. The requester always requests the VMEbus at level 3 the first time following a SYSRESET.

0 VMEbus request level 0

1 VMEbus request level 1

2 VMEbus request level 2

3 VMEbus request level 3

DRELM DMAC Release Mode. These bits define the VMEbus release mode for the DMAC requester. The DMAC always releases the bus when the FIFO is full (VMEbus to local peripheral bus) or empty (local peripheral bus to VMEbus).

0 Release when the time on timer has expired and a BRx* signal is active on the VMEbus.

1 Release when the time on timer has expired.

2 Release when a BRx* signal is active on the VMEbus.

3 Release when a BRx* signal is active on the VMEbus or the time on timer has expired.

DFAIR DMAC Fair Mode. When this bit is high, the DMAC requester operates in the fair mode. It waits until its request level is inactive before requesting the VMEbus. When this bit is low, the DMAC requester does not operate in the fair mode.

DTBL DMAC Direct Mode. The DMAC operates in the direct mode when this bit is low, and it operates in the command chaining mode when this bit is high.

DEN DMAC Enable. The DMAC is enabled when this bit is set high. This bit always reads 0.

DHALT DMAC Halt. When this bit is high, the DMAC halts at the end of a command when the DMAC is operating in the command chaining mode. When this bit is low, the DMAC executes the next command in the list.

DMAC Control Register 2 (bits 8-15)

REG	DMAC Control Register 2							
ADR/SIZ	$FFF40034 (8 bits [7 used] of 32)							
BIT	15	14	13	12	11	10	0	8
FIELD	INTE	SNP			VINC	LINC	TVME	D16
OPER	R/W	R/W			R/W	R/W	R/W	R/W
RESET	0 PS	0 PS			0 PS	0 PS	0 PS	0 PS

This portion of the control register is loaded by the processor or by the DMAC when it loads the command word from the command packet. Because this register is loaded from the command packet in the command chaining mode, the descriptions here also apply to the control word in the command packet.

D16 DMAC D16 Access. When this bit is high, the DMAC executes D16 cycles on the VMEbus. When this bit is low, the DMAC executes D32 cycles on the VMEbus.

TVME DMAC Transfer to VMEbus. This bit defines the direction in which the DMAC transfers data. When this bit is high, data is transferred to the VMEbus. When it is low, data is transferred to the local peripheral bus.

LINC Local Peripheral Bus Increment. When this bit is high, the local peripheral bus address counter is incremented during DMA transfers. When this bit is low, the counter is not incremented. This bit should normally be set high. In special situations such as transferring data to or from a FIFO, it may be desirable to not increment the counter.

VINC VMEbus Increment. When this bit is high, the VMEbus address counter is incremented during DMA transfers. When this bit is low, the counter is not incremented. This bit should normally be set high. In special situations such as transferring data to or from a FIFO, it may be desirable to not increment the counter.

SNP DMAC Snoop Mode. (This feature is not applicable to any of the MVME197 module series).

INTE DMAC Interrupt. This bit is used only in the command chaining mode and it is only modified when the DMAC loads the control register from the control word in the command packet. When this bit in the command packet is set, an interrupt is sent to the local peripheral bus interrupter when the command in the packet has been executed. The local peripheral bus is interrupted if the DMAC interrupt is enabled.

DMAC Control Register 2 (bits 0-7)

REG	DMAC Control Register 2							
ADR/SIZ	$FFF40034 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD	BLK		VME AM					
OPER	R/W		R/W					
RESET	0 PS		0 PS					

This portion of the control register is loaded by the processor or the DMAC when it loads the command word from the command packet. Because this byte is loaded from the command packet in the command chaining mode, the descriptions here also apply to the control word in the command packet.

VME AM VMEbus Address Modifier. These bits define the address modifier codes the DMAC drives on the VMEbus when it is bus master. During non-block transfer cycles, bits 0-5 define the VMEbus address modifiers. During block transfers, bits 0 and 1 are modified by the DMAC to indicate a block transfer. Block transfer mode should not be set in the address modifier codes. The special block transfer bits should be set to enable block transfers. If non-block cycles are required to reach a 32- or 64-bit boundary, bits 0 and 1 are used during these cycles.

BLK DMAC Block Transfer. These bits control the block transfer modes of the DMAC:

0 Block transfers disabled.

1 The DMAC executes D32 block transfer cycles on the VMEbus. In the block transfer mode, the DMAC may execute byte and two-byte cycles at the beginning and ending of a transfer in non-block transfer mode.

2 Block transfers disabled.

3 The DMAC executes D64 block transfer cycles on the VMEbus. In the block transfer mode, the DMAC may execute byte, two-byte and four-byte cycles at the beginning and ending of a transfer in non-block transfer mode.

DMAC Local Peripheral Bus Address Counter Register

REG	DMAC Local Peripheral Bus Address Counter Register					
ADR/SIZ	$FFF40038 (32 bits)					
BIT	31	------------------	16	15	------------------	0
FIELD	Local Peripheral Bus Address Counter					
OPER	R/W					
RESET	0 PS					

In the direct mode, this counter is programmed with the starting address of the data in local peripheral bus.

DMAC VMEbus Address Counter Register

REG	DMAC VMEbus Address Counter Register					
ADR/SIZ	$FFF4003C (32 bits)					
BIT	31	------------------	16	15	------------------	0
FIELD	VMEbus Address Counter					
OPER	R/W					
RESET	0 PS					

In the direct mode, this counter is programmed with the starting address of the data in VMEbus memory.

DMAC Byte Counter Register

REG	DMAC Byte Counter Register					
ADR/SIZ	$FFF40040 (32 bits)					
BIT	31	------------------	16	15	------------------	0
FIELD	Byte Counter					
OPER	R/W					
RESET	0 PS					

In the direct mode, this counter is programmed with the number of bytes of data to be transferred.

Table Address Counter Register

REG	Table Address Counter Register					
ADR/SIZ	$FFF40044 (32 bits)					
BIT	31	------------------	16	15	------------------	0
FIELD	Table Address Counter					
OPER	R/W					
RESET	0 PS					

In the command chaining mode, this counter should be loaded by the processor with the starting address of the list of commands. This register gets reloaded by the DMAC with the starting address of the current command. The last command in a list should have bits 0 and 1 set in the next command pointer.

VMEbus Interrupter Control Register

REG	VMEbus Interrupter Control Register							
ADR/SIZ	$FFF40048 (8 bits [7 used] of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD		IRQ1S		IRQC	IRQS	IRQL		
OPER		R/W		S	R	S		
RESET		0 PS		0 PS	0 PS	0 PS		

This register controls the VMEbus interrupter.

IRQL VMEbus Interrupt Level. These bits define the level of the VMEbus interrupt generated by the VMEchip2. A VMEbus interrupt is generated by writing the desired level to these bits. These bits always read 0 and writing 0 to these bits has no effect.

IRQS Interrupt Status. This bit is the IRQ status bit. When this bit is high, the VMEbus interrupt has not been acknowledged. When this bit is low, the VMEbus interrupt has been acknowledged. This is a read-only status bit.

IRQC Interrupt Clear. This bit is VMEbus interrupt clear bit. When this bit is set high, the VMEbus interrupt is removed. This feature is only used when the IRQ1 broadcast mode is used. Normal VMEbus interrupts should never be cleared. This bit always reads 0 and writing a 0 to this bit has no effect.

IRQ1S Interrupt 1 Signal. These bits control the function of the IRQ1 signal line on the VMEbus:

0 The IRQ1 signal from the interrupter is connected to the IRQ1 signal line on the VMEbus.

1 The output from tick timer 1 is connected to the IRQ1 signal line on the VMEbus.

2 The IRQ1 signal from the interrupter is connected to the IRQ1 signal line on the VMEbus.

3 The output from tick timer 2 is connected to the IRQ1 signal line on the VMEbus.

VMEbus Interrupter Vector Register

REG	VMEbus Interrupter Vector Register							
ADR/SIZ	$FFF40048 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD	Interrupter Vector							
OPER	R/W							
RESET	$0F PS							

This register controls the VMEbus interrupter vector.

MPU Status and DMAC Interrupt Count Register

REG	MPU Status and DMAC Interrupt Count Register							
ADR/SIZ	$FFF40048 (8 bits of 32)							
BIT	15	14	13	12	11	10	0	8
FIELD	DMAIC				MCLR	MLBE	MLPE	MLOB
OPER	R				S	R	R	R
RESET	0 PS				0 PS	0 PS	0 PS	0 PS

This is the MPU status register (the MPU status bits are not applicable to the MVME197 module series) and DMAC interrupt counter.

MLOB MPU Offboard. When this bit is set, the MPU received a TEA and the status indicated offboard. This bit is cleared by writing a one to the MCLR bit.

MLPE MPU Parity Error. When this bit is set, the MPU received a TEA and the status indicated a parity error during a DRAM data transfer. This bit is cleared by writing a one to the MCLR bit.

MLBE MPU Bus Enable. When this bit is set, the MPU received a TEA and additional status was not provided. This bit is cleared by writing a one to the MCLR bit.

MCLR MPU Clear. Writing a one to this bit clears the MPU status bits MLTO, MLOB, MLPE, and MLBE.

DMAIC DMAC Interrupt Counter. The DMAC interrupt counter is incremented when an interrupt is sent to the local peripheral bus interrupter. The value in this counter indicates the number of commands processed when the DMAC is operated in the command chaining mode. If interrupt count exceeds 15, the counter rolls over. This counter operates regardless of whether the DMAC interrupts are enabled.

DMAC Status Register

REG	DMAC Status Register							
ADR/SIZ	$FFF40048 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD	MLTO	DLBE	DLPE	DLOB	DLTO	TBL	VME	DONE
OPER	R	R	R	R	R	R	R	R
RESET	0 PS	0 PS	0 PS	0 PS	0 PS	0 PS	0 PS	0 PS

This is the DMAC status register.

DONE DMAC Done. This bit is set when the DMAC has finished executing commands and there were no errors or the DMAC has finished executing command because the halt bit was set. This bit is cleared when the DMAC is enabled.

VME DMAC VMEbus Bus Error. When this bit is set, the DMAC received a VMEbus BERR during a data transfer. This bit is cleared when the DMAC is enabled.

TBL DMAC Local Peripheral Bus Error. When this bit is set, the DMAC received an error on the local peripheral bus while it was reading commands from the command packet. Additional information is provided in bits 3 - 6 (DLTO, DLOB, DLPE, and DLBE). This bit is cleared when the DMAC is enabled.

DLTO DMAC Local Peripheral Bus Timeout. When this bit is set, the DMAC received a TEA and the status indicated a local peripheral bus timeout. This bit is cleared when the DMAC is enabled.

DLOB DMAC Local Peripheral Bus Offboard. When this bit is set, the DMAC received a TEA and the status indicated offboard. This bit is cleared when the DMAC is enabled.

DLPE DMAC Local Peripheral Bus Parity Error. When this bit is set, the DMAC received a TEA and the status indicated a parity error during a DRAM data transfer. This bit is cleared when the DMAC is enabled.

DLBE DMAC Local Peripheral Bus Enable. When this bit is set, the DMAC received a TEA and additional status was not provided. This bit is cleared when the DMAC is enabled.

MLTO MPU Local Peripheral Bus Timeout. When this bit is set, the MC68040 bus master received a TEA and the status indicated a local peripheral bus timeout. This bit is cleared by a writing a one to the MCLR bit.

Programming the Tick and Watchdog Timers

The VMEchip2 has two 32-bit tick timers and one watchdog timer. This section provides addresses and bit level descriptions of the prescaler, tick timer, watchdog timer registers and various other timer registers.

VMEbus Arbiter Timeout Control Register

REG	VMEbus Arbiter Timeout Control Register							
ADR/SIZ	$FFF4004C (8 bits [1 used] of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD								ARBTO
OPER								R/W
RESET								0 PS

This register controls the VMEbus arbiter timeout timer.

ARBTO VMEbus Arbiter Timeout. When this bit is high, the VMEbus grant timeout timer is enabled. When this bit is low, the VMEbus grant timer is disabled. When the timer is enabled and the arbiter does not receive a BBSY signal within 256 µsec after a grant is issued, the arbiter asserts BBSY and removes the grant. The arbiter then re-arbitrates any pending requests.

DMAC Ton/Toff Timers and VMEbus Global Timeout Control Register

REG	DMAC Ton/Toff Timers and VMEbus Global Timeout Control Register							
ADR/SIZ	$FFF4004C (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD	TIME OFF			TIME ON			VGTO	
OPER	R/W			R/W			R/W	
RESET	0 PS			0 PS			0 PS	

This register controls the DMAC time off timer, the DMAC time on timer, and the VMEbus global timeout timer.

VGTO VMEbus Global Timeout. These bits define the VMEbus global timeout value. When DS0 or DS1 is asserted on the VMEbus, the timer begins timing. If the timer times out before the data strobes are removed, a BERR signal is sent to the VMEbus. The global timeout timer is disabled when the VMEchip2 is not system controller.

0	8 µsec
1	64 µsec
2	256 µsec
3	The timer is disabled.

TIME ON DMAC Time On. These bits define the maximum time the DMAC spends on the VMEbus:

0	16 µsec
1	32 µsec
2	64 µsec
3	128 µsec
4	256 µsec
5	512 µsec
6	1024 µsec
7	When done (or no data)

TIME OFF DMAC Time Off. These bits define the minimum time the DMAC spends off the VMEbus:

0	0 µsec
1	16 µsec
2	32 µsec
3	64 µsec
4	128 µsec
5	256 µsec
6	512 µsec
7	1024 µsec

VME Access, Local Peripheral Bus and Watchdog Timeout Control Register

REG	VME Access, Local Peripheral Bus and Watchdog Timeout Control Register							
ADR/SIZ	$FFF4004C (8 bits of 32)							
BIT	15	14	13	12	11	10	0	8
FIELD	VATO		LBTO		WDTO			
OPER	R/W		R/W		R/W			
RESET	0 PS		0 PS		0 PS			

WDTO Watchdog Timeout. These bits define the watchdog timeout period:

0	512 µsec
1	1 msec
2	2 msec
3	4 msec
4	8 msec
5	16 msec
6	32 msec
7	64 msec
8	128 msec
9	256 msec
10	512 msec
11	1 sec
12	4 sec

13	16 sec
14	32 sec
15	64 sec

LBTO Local Peripheral Bus Timeout. These bits define the local peripheral bus timeout value. The timer begins timing when TS is asserted on the local peripheral bus. If TA or TAE is not asserted before the timer times out, a TEA signal is sent to the local peripheral bus. The timer is disabled if the transfer is bound for the VMEbus.

0	8 µsec
1	64 µsec
2	256 µsec
3	The timer is disabled.

VATO VMEbus Access Timeout. These bits define the VMEbus access timeout value. When a transaction is headed to the VMEbus and the VMEchip2 is not the current VMEbus master, the access timer begins timing. If the VMEchip2 has not received bus mastership before the timer times out and the transaction is not write posted, a TEA signal is sent to the local peripheral bus. If the transaction is write posted, a write post error interrupt is sent to the local peripheral bus interrupter.

0	64 µsec
1	1 msec
2	32 msec
3	The timer is disabled.

Prescaler Control Register

REG	Prescaler Control Register							
ADR/SIZ	$FFF4004C (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD	Prescaler Adjust							
OPER	R/W							
RESET	$DF P							

The prescaler provides the various clocks required by the counters and timers in the VMEchip2. In order to specify absolute times from these counters and timers, the prescaler must be adjusted for different bus clocks. The prescaler register should be programmed based on the following equation. This provides a one MHz clock to the Tick timers.

Prescaler register – 256 - B clock (MHz)

For example, for operation at 20 MHz the prescaler value is $EC, at 25 MHz it is $E7, and at 33 MHz it is $DF. Notice that the B clock here is half of the frequency of the MC88110 speed in the board.

Non-integer bus clocks introduce an error into the specified times for the various counters and timers. This is most notable in the tick timers. The tick timer clock can be derived by the following equation.

Tick timer clk = B clock / (256 - Prescaler Value)

If the prescaler is not correctly programmed, the bus timers do not generate their specified values and the VMEbus reset time may be violated. The maximum clock frequency for the tick timers is the B clock divided by two. The prescaler register control logic does not allow the value 255 ($FF) to be programmed.

Tick Timer 1 Compare Register

REG	Tick Timer 1 Compare Register					
ADR/SIZ	$FFF40050 (32 bits)					
BIT	31	------------------	16	15	------------------	0
FIELD	Tick Timer 1 Compare					
OPER	R/W					
RESET	0 PS					

The tick timer 1 counter is compared to this register. When they are equal, an interrupt is sent to the local peripheral bus interrupter and the overflow counter is incremented. If the clear-on-compare mode is enabled, the counter is also cleared. For periodic interrupts, the following equation should be used to calculate the compare register value for a specific period (T).

$$\text{Compare Register Value} = T\ (\mu\text{sec})$$

When programming the tick timer for periodic interrupts, the counter should be cleared to zero by software and then enabled. If the counter does not initially start at zero, the time to the first interrupt may be longer or shorter than expected. Remember the roll over time for the counter is 71.6 minutes.

Tick Timer 1 Counter Register

REG	Tick Timer 1 Counter Register					
ADR/SIZ	$FFF40054 (32 bits)					
BIT	31	------------------	16	15	------------------	0
FIELD	Tick Timer 1 Counter					
OPER	R/W					
RESET	0 PS					

This is the tick timer 1 counter. When enabled, it increments every microsecond. Software may read or write the counter at any time.

Tick Timer 2 Compare Register

REG	Tick Timer 2 Compare Register					
ADR/SIZ	$FFF40058 (32 bits)					
BIT	31	------------------	16	15	------------------	0
FIELD	Tick Timer 2 Compare					
OPER	R/W					
RESET	0 P					

The tick timer 2 counter is compared to this register. When they are equal, an interrupt is sent to the local peripheral bus interrupter and the overflow counter is incremented. If the clear-on-compare mode is enabled, the counter is also cleared. For periodic interrupts, the following equation should be used to calculate the compare register value for a specific period (T).

$$\text{Compare Register Value} = \text{T } (\mu\text{sec})$$

When programming the tick timer for periodic interrupts, the counter should be cleared to zero by software and then enabled. If the counter does not initially start at zero, the time to the first interrupt may be longer or shorter than expected. Remember the roll over time for the counter is 71.6 minutes.

Tick Timer 2 Counter Register

REG	Tick Timer 2 Counter Register					
ADR/SIZ	$FFF4005C (32 bits)					
BIT	31	------------------	16	15	------------------	0
FIELD	Tick Timer 2 Counter					
OPER	R/W					
RESET	0 P					

This is the tick timer 2 counter. When enabled, it increments every microsecond. Software may read or write the counter at any time.

Board Control Register

REG	Board Control Register							
ADR/SIZ	$FFF40060 (8 bits [7 used] of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD		SCON	SFFL	BRFLI	PURS	CPURS	BDFLO	RSWE
OPER		R	R	R	R	C	R/W	R/W
RESET		X	X	1 PSL	1 P	0 PS	1 PSL	1 P

RSWE Reset Switch Enable. When this bit is high, the RESET switch is enabled to the VMEchip2. When this bit is low, the RESET switch is disabled to the VMEchip2. Since the BusSwitch also receives RESET switch, RESET is still enabled even if the register disables RESET to the VMEchip2.

BDFLO Board Fail Output. When this bit is high, the VMEchip2 asserts the BRDFAIL signal pin. When this bit is low, this bit does not contribute to the BRDFAIL signal on the VMEchip2.

CPURS Clear Power-Up Status. When this bit is set high, the power-up reset status bit is cleared. This bit is always read zero.

PURS Power-Up Reset Status. This bit is set by a power-up reset. It is cleared by a write to the CPURS bit.

BRFLI Board Fail Status. When this status bit is high, the BRDFAIL signal pin on the VMEchip2 is asserted. When this status bit is low, the BRDFAIL signal pin on the VMEchip2 is not asserted. The BRDFAIL pin may be asserted by an external device, the BDFLO bit in this register, or a watchdog timeout.

SFFL System Fail. When this status bit is high, the SYSFAIL signal line on the VMEbus is asserted. When this status bit is low, the SYSFAIL signal line on the VMEbus is not asserted.

SCON System Controller. When this status bit is high, the VMEchip2 is configured as system controller. When this status bit is low, the VMEchip2 is not configured as system controller.

Watchdog Timer Control Register

REG	Watchdog Timer Control Register							
ADR/SIZ	$FFF40060 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD	SRST	WDCS	WDCC	WDTO	WDBFE	WDS/L	WDRSE	WDEN
OPER	S	C	C	R	R/W	R/W	R/W	R/W
RESET	0 PS	0	0	0 P	0 PSL	0 PSL	0 PSL	0 PSL

WDEN Watchdog Enable. When this bit is high, the watchdog timer is enabled. When this bit is low, the watchdog timer is not enabled.

WDRSE Watchdog Reset Enable. When this bit is high, and a watchdog timeout occurs, a SYSRESET or LRESET is generated. The WDS/L bit in this register selects the reset. When this bit is low, a watchdog timeout does not cause a reset.

WDS/L Watchdog System/Local Reset. When this bit is high and the watchdog timer has timed out and the watchdog reset enable (WDRSE bit in this register) is high, a SYSRESET signal is generated on the VMEbus which in turn causes LRESET to be asserted. When this bit is low and the watchdog timer has timed out and the watchdog reset enable (WDRSE bit in this register) is high, an LRESET signal from the VMEchip2 is generated.

WDBFE Watchdog Board Fail Enable. When this bit is high and the watchdog timer has timed out, the VMEchip2 asserts the BRDFAIL signal pin. When this bit is low, the watchdog timer does not contribute to the BRDFAIL signal on the VMEchip2.

WDTO Watchdog Timeout Status. When this status bit is high, a watchdog timeout has occurred. When this status bit is low, a watchdog timeout has not occurred. This bit is cleared by writing a one to the WDCS bit in this register.

WDCC Watchdog Clear Counter. When this bit is set high, the watchdog counter is reset. The counter must be reset within the timeout period or a watchdog timeout occurs.

WDCS Watchdog Clear Status. When this bit is set high, the watchdog timeout status bit (WDTO bit in this register) is cleared.

SRST System Reset. When this bit is set high, a SYSRESET signal is generated on the VMEbus. SYSRESET resets the VMEchip2 and clears this bit.

Tick Timer 2 Control Register

REG	Tick Timer 2 Control Register							
ADR/SIZ	$FFF40060 (8 bits [7 used] of 32)							
BIT	15	14	13	12	11	10	0	8
FIELD	OVF3	OVF2	OVF1	OVF0		COVF	COC	EN
OPER	R	R	R	R		C	R/W	R/W
RESET	0 PS	0 PS	0 PS	0 PS		0 PS	0 PS	0 PS

EN Counter Enable. When this bit is high, the counter increments. When this bit is low, the counter does not increment.

COC Clear On Compare 2. When this bit is high, the counter is reset to zero when it compares with the compare register. When this bit is low, the counter is not reset.

COVF Clear Overflow Counter 2. The overflow counter is cleared when a one is written to this bit.

OVF0-3 Overflow Counter (0 - 3). These four bits are the output of the overflow counter. The overflow counter is incremented each time the tick timer sends an interrupt to the local peripheral bus interrupter. The overflow counter can be cleared by writing a one to the COVF bit.

Tick Timer 1 Control Register

REG	Tick Timer 1 Control Register							
ADR/SIZ	$FFF40060 (8 bits [7 used] of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD	OVF3	OVF2	OVF1	OVF0		COVF	COC	EN
OPER	R	R	R	R		C	R/W	R/W
RESET	0 PS	0 PS	0 PS	0 PS		0 PS	0 PS	0 PS

EN Counter Enable. When this bit is high, the counter increments. When this bit is low, the counter does not increment.

COC Clear On Compare 1. When this bit is high, the counter is reset to zero when it compares with the compare register. When this bit is low, the counter is not reset.

COVF Clear Overflow Counter 1. The overflow counter is cleared when a one is written to this bit.

OVF3-0 Overflow Counter (3 - 0). These four bits are the output of the overflow counter. The overflow counter is incremented each time the tick timer sends an interrupt to the local peripheral bus interrupter. The overflow counter can be cleared by writing a one to the COVF bit.

Prescaler Counter Register

REG	Prescaler Counter Register					
ADR/SIZ	$FFF40064 (32 bits)					
BIT	31	-------------------	16	15	-------------------	0
FIELD	Prescaler Counter					
OPER	R/W					
RESET	0 P					

The VMEchip2 has a 32-bit prescaler that provides the clocks required by the various timers in the chip. Access to the prescaler is provided for test purposes. The counter is described here because it may be useful in other applications. The lower 8 bits of the prescaler counter increment to $FF at the bus clock rate and then they are loaded from the prescaler adjust register. When the load occurs, the upper 24 bits are incremented. When the prescaler adjust register is correctly programmed, the lower 8 bits increment at the bus clock rate and the upper 24 bits increment every microsecond. The counter may be read at any time.

Programming the Local Peripheral Bus Interrupter

The local peripheral bus interrupter is used by devices that wish to interrupt the local peripheral bus. There are 31 devices that can interrupt the local peripheral bus through the VMEchip2. In the general case, each interrupter has a level select register, an enable bit, a status bit, a clear bit, and for the software interrupts, a set bit. Each interrupter also provides a unique interrupt vector to the processor. The upper four bits of the vector are programmable in the vector base registers. The lower four bits are unique for each interrupter. There are two base registers, one for the first 16 interrupters, and one for the next 8 interrupters. The VMEbus interrupters provide their own vectors. A summary of the interrupts is shown in Table 2-3.

The status bit of an interrupter is affected by the enable bit. If the enable bit is low, the status bit is also low. Interrupts may be polled by setting the enable bit and programming the level to zero. This enables the status bit and prevents the local peripheral bus from being interrupted. The enable bit does not clear edge-sensitive interrupts. If necessary, edge-sensitive interrupts should be cleared, in order to remove any old interrupts, and then enabled. The master interrupt enable (MIEN) bit must be set before the VMEchip2 can generate any interrupts. The MIEN bit is in the I/O Control Register 1.

Table 2-3. Local Peripheral Bus Interrupter Summary

Interrupt	Vector	Priority for Simultaneous Interrupts
VMEbus IRQ1	External	Lowest
VMEbus IRQ2	External	
VMEbus IRQ3	External	
VMEbus IRQ4	External	
VMEbus IRQ5	External	
VMEbus IRQ6	External	
VMEbus IRQ7	External	
Spare	$Y7	
Software 0	$Y8	
Software 1	$Y9	
Software 2	$YA	
Software 3	$YB	
Software 4	$YC	
Software 5	$YD	
Software 6	$YE	
Software 7	$YF	
GCSR LM0	$X0	
GCSR LM1	$X1	
GCSR SIG0	$X2	
GCSR SIG1	$X3	
GCSR SIG2	$X4	
GCSR SIG3	$X5	
DMAC	$X6	
VMEbus Interrupter Acknowledge	$X7	
Tick Timer 1	$X8	
Tick Timer 2	$X9	
VMEbus IRQ1 Edge-Sensitive	$XA	
External Input (parity error)	$XB	
VMEbus Master Write Post Error	$XC	
VMEbus SYSFAIL	$XD	
Abort Switch	$XE	
VMEbus ACFAIL	$XF	Highest

Notes

X = The contents of vector base register 0.

Y = The contents of vector base register 1.

Refer to the Vector Base Register description later in this chapter for recommended Vector Base Register values.

Local Peripheral Bus Interrupter Status Register (bits 24-31)

REG	Local Peripheral Bus Interrupter Status Register (bits 24-31)							
ADR/SIZ	$FFF40068 (8 bits of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD	ACF	AB	SYSF	MWP	PE	VI1E	TIC2	TIC1
OPER	R	R	R	R	R	R	R	R
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the local peripheral bus interrupter status register. When an interrupt status bit is high, a local peripheral bus interrupt is being generated. When an interrupt status bit is low, a local interrupt is not being generated. The interrupt status bits are:

TIC1 Tick Timer 1 Interrupt.

TIC2 Tick Timer 2 Interrupt.

VI1E VMEbus IRQ1 Edge-Sensitive Interrupt.

PE External Interrupt (Parity Error).

MWP VMEbus Master Write Post Error Interrupt.

SYSF VMEbus SYSFAIL Interrupt.

AB ABORT Switch Interrupt.

ACF VMEbus ACFAIL Interrupt.

Local Peripheral Bus Interrupter Status Register (bits 16-23)

REG	Local Peripheral Bus Interrupter Status Register (bits 16-23)							
ADR/SIZ	$FFF40068 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD	VIA	DMA	SIG3	SIG2	SIG1	SIG0	LM1	LM0
OPER	R	R	R	R	R	R	R	R
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the local peripheral bus interrupter status register. When an interrupt status bit is high, a local peripheral bus interrupt is being generated. When an interrupt status bit is low, a local interrupt is not being generated. The interrupt status bits are:

LMO	GCSR LM0 Interrupt.
LM1	GCSR LM1 Interrupt.
SIG0	GCSR SIG0 Interrupt.
SIG1	GCSR SIG1 Interrupt.
SIG2	GCSR SIG2 Interrupt.
SIG3	GCSR SIG3 Interrupt.
DMA	DMAC Interrupt.
VIA	VMEbus Interrupter Acknowledge Interrupt.

Local Peripheral Bus Interrupter Status Register (bits 8-15)

REG	Local Peripheral Bus Interrupter Status Register (bits 8-15)							
ADR/SIZ	$FFF40068 (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
FIELD	SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
OPER	R	R	R	R	R	R	R	R
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the local peripheral bus interrupter status register. When an interrupt status bit is high, a local peripheral bus interrupt is being generated. When an interrupt status bit is low, a local interrupt is not being generated. The interrupt status bits are:

SW0 Software 0 Interrupt.

SW1 Software 1 Interrupt.

SW2 Software 2 Interrupt.

SW3 Software 3 Interrupt.

SW4 Software 4 Interrupt.

SW5 Software 5 Interrupt.

SW6 Software 6 Interrupt.

SW7 Software 7 Interrupt.

Local Peripheral Bus Interrupter Status Register (bits 0-7)

REG	Local Peripheral Bus Interrupter Status Register (bits 0-7)							
ADR/SIZ	$FFF40068 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD	SPARE	VME7	VME6	VME5	VME4	VME3	VME2	VME1
OPER	R	R	R	R	R	R	R	R
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the local peripheral bus interrupter status register. When an interrupt status bit is high, a local peripheral bus interrupt is being generated. When an interrupt status bit is low, a local interrupt is not being generated. The interrupt status bits are:

VME1 VMEbus IRQ1 Interrupt.

VME2 VMEbus IRQ2 Interrupt.

VME3 VMEbus IRQ3 Interrupt.

VME4 VMEbus IRQ4 Interrupt.

VME5 VMEbus IRQ5 Interrupt.

VME6 VMEbus IRQ6 Interrupt.

VME7 VMEbus IRQ7 Interrupt.

SPARE This bit is not used.

Local Peripheral Bus Interrupter Enable Register (bits 24-31)

REG	Local Peripheral Bus Interrupter Enable Register (bits 24-31)							
ADR/SIZ	$FFF4006C (8 bits of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD	EACF	EAB	ESYSF	EMWP	EPE	EVI1E	ETIC2	ETIC1
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the local peripheral bus interrupter enable register. When an enable bit is high, the corresponding interrupt is enabled. When an enable bit is low, the corresponding interrupt is disabled. The enable bit does not clear edge-sensitive interrupts or prevent the flip flop from being set. If necessary, edge-sensitive interrupters should be cleared to remove any old interrupts and then enabled.

ETIC1 Enable Tick Timer 1 Interrupt.

ETIC2 Enable Tick Timer 2 Interrupt.

EVI1E Enable VMEbus IRQ1 Edge-Sensitive Interrupt.

EPE Enable External Interrupt (Parity Error).

EMWP Enable VMEbus Master Write Post Error Interrupt.

ESYSF Enable VMEbus SYSFAIL Interrupt.

EAB Enable ABORT Switch Interrupt.

EACF Enable VMEbus ACFAIL Interrupt.

Local Peripheral Bus Interrupter Enable Register (bits 16-23)

REG	Local Peripheral Bus Interrupter Enable Register (bits 16-23)							
ADR/SIZ	$FFF4006C (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD	EVIA	EDMA	ESIG3	ESIG2	ESIG1	ESIG0	ELM1	ELM0
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the local peripheral bus interrupter enable register. When an enable bit is high, the corresponding interrupt is enabled. When an enable bit is low, the corresponding interrupt is disabled. The enable bit does not clear edge-sensitive interrupts or prevent the flip flop from being set. If necessary, edge-sensitive interrupters should be cleared to remove any old interrupts and then enabled.

ELMO Enable GCSR LM0 Interrupt.

ELM1 Enable GCSR LM1 Interrupt.

ESIG0 Enable GCSR SIG0 Interrupt.

ESIG1 Enable GCSR SIG1 Interrupt.

ESIG2 Enable GCSR SIG2 Interrupt.

ESIG3 Enable GCSR SIG3 Interrupt.

EDMA Enable DMAC Interrupt.

EVIA VMEbus Interrupter Acknowledge Interrupt.

Local Peripheral Bus Interrupter Enable Register (bits 8-15)

REG	Local Peripheral Bus Interrupter Enable Register (bits 8-15)							
ADR/SIZ	$FFF4006C (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
FIELD	ESW7	ESW6	ESW5	ESW4	ESW3	ESW2	ESW1	ESW0
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This is the local peripheral bus interrupter enable register. When an enable bit is high, the corresponding interrupt is enabled. When an enable bit is low, the corresponding interrupt is disabled. The enable bit does not clear edge-sensitive interrupts or prevent the flip flop from being set. If necessary, edge-sensitive interrupters should be cleared to remove any old interrupts and then enabled.

ESW0 Enable Software 0 Interrupt.

ESW1 Enable Software 1 Interrupt.

ESW2 Enable Software 2 Interrupt.

ESW3 Enable Software 3 Interrupt.

ESW4 Enable Software 4 Interrupt.

ESW5 Enable Software 5 Interrupt.

ESW6 Enable Software 6 Interrupt.

ESW7 Enable Software 7 Interrupt.

Local Peripheral Bus Interrupter Enable Register (bits 0-7)

REG	Local Peripheral Bus Interrupter Enable Register (bits 0-7)							
ADR/SIZ	$FFF4006C (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD	SPARE	EIRQ7	EIRQ6	EIRQ5	EIRQ4	EIRQ3	EIRQ2	EIRQ1
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This is the local peripheral bus interrupter enable register. When an enable bit is high, the corresponding interrupt is enabled. When an enable bit is low, the corresponding interrupt is disabled. The enable bit does not clear edge-sensitive interrupts or prevent the flip flop from being set. If necessary, edge-sensitive interrupters should be cleared to remove any old interrupts and then enabled.

EIRQ1 Enable VMEbus IRQ1 Interrupt.

EIRQ2 Enable VMEbus IRQ2 Interrupt.

EIRQ3 Enable VMEbus IRQ3 Interrupt.

EIRQ4 Enable VMEbus IRQ4 Interrupt.

EIRQ5 Enable VMEbus IRQ5 Interrupt.

EIRQ6 Enable VMEbus IRQ6 Interrupt.

EIRQ7 Enable VMEbus IRQ7 Interrupt.

SPARE This bit is not used.

Software Interrupt Set Register (bits 8-15)

REG	Software Interrupt Set Register (bits 8-15)							
ADR/SIZ	$FFF40070 (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
FIELD	SSW7	SSW6	SSW5	SSW4	SSW3	SSW2	SSW1	SSW0
OPER	S	S	S	S	S	S	S	S
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is used to set the software interrupts. An interrupt is set by writing a one to it. The software interrupt set bits are:

SSW0 Set Software 0 Interrupt.

SSW1 Set Software 1 Interrupt.

SSW2 Set Software 2 Interrupt.

SSW3 Set Software 3 Interrupt.

SSW4 Set Software 4 Interrupt.

SSW5 Set Software 5 Interrupt.

SSW6 Set Software 6 Interrupt.

SSW7 Set Software 7 Interrupt.

Interrupt Clear Register (bits 24-31)

REG	Interrupt Clear Register (bits 24-31)							
ADR/SIZ	$FFF40074 (8 bits of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD	CACF	CAB	CSYSF	CMWP	CPE	CVI1E	CTIC2	CTIC1
OPER	C	C	C	C	C	C	C	C
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is used to clear the edge-sensitive interrupts. An interrupt is cleared by writing a one to its clear bit. The clear bits are defined below.

CTIC1 Clear Tick Timer 1 Interrupt.

CTIC2 Clear Tick Timer 2 Interrupt.

CVI1E Clear VMEbus IRQ1 Edge-Sensitive Interrupt.

CPE Clear External Interrupt (Parity Error).

CMWP Clear VMEbus Master Write Post Error Interrupt.

CSYSF Clear VMEbus SYSFAIL Interrupt.

CAB Clear ABORT Switch Interrupt.

CACF Clear VMEbus ACFAIL Interrupt.

Interrupt Clear Register (bits 16-23)

REG	Interrupt Clear Register (bits 16-23)							
ADR/SIZ	$FFF40074 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD	CVIA	CDMA	CSIG3	CSIG2	CSIG1	CSIG0	CLM1	CLM0
OPER	C	C	C	C	C	C	C	C
RESET	X	X	X	X	X	X	X	X

This register is used to clear the edge sensitive-interrupts. An interrupt is cleared by writing a one to its clear bit. The clear bits are defined below.

CLMO Clear GCSR LM0 Interrupt.

CLM1 Clear GCSR LM1 Interrupt.

CSIG0 Clear GCSR SIG0 Interrupt.

CSIG1 Clear GCSR SIG1 Interrupt.

CSIG2 Clear GCSR SIG2 Interrupt.

CSIG3 Clear GCSR SIG3 Interrupt.

CDMA Clear DMA Controller Interrupt.

CVIA Clear VMEbus Interrupter Acknowledge Interrupt.

Interrupt Clear Register (bits 8-15)

REG	Interrupt Clear Register (bits 8-15)							
ADR/SIZ	$FFF40074 (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
FIELD	CSW7	CSW6	CSW5	CSW4	CSW3	CSW72	CSW1	CSW0
OPER	C	C	C	C	C	C	C	C
RESET	X	X	X	X	X	X	X	X

This register is used to clear the edge software interrupts. An interrupt is cleared by writing a one to its clear bit. The clear bits are:

CSW0	Clear Software 0 Interrupt.
CSW1	Clear Software 1 Interrupt.
CSW2	Clear Software 2 Interrupt.
CSW3	Clear Software 3 Interrupt.
CSW4	Clear Software 4 Interrupt.
CSW5	Clear Software 5 Interrupt.
CSW6	Clear Software 6 Interrupt.
CSW7	Clear Software 7 Interrupt.

Interrupt Level Register 1 (bits 24-31)

REG	Interrupt Level Register 1 (bits 24-31)							
ADR/SIZ	$FFF40078 (8 bits [6 used] of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD		ACF LEVEL				AB LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the abort interrupt and the ACFAIL interrupt.

AB LEVEL Abort Interrupt Level. These three bits define the level of the abort interrupt. Note that on the MVME197 module series, ABORT is handled by the BusSwitch to generate NMI* to the processor.

ACF LEVEL ACFAIL Interrupt Level. These three bits define the level of the ACFAIL interrupt.

Interrupt Level Register 1 (bits 16-23)

REG	Interrupt Level Register 1 (bits 16-23)							
ADR/SIZ	$FFF40078 (8 bits [6 used] of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD		SYSF LEVEL				WPE LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the SYSFAIL interrupt and the master write post bus error interrupt.

WPE LEVEL Write Post Error Interrupt Level. These three bits define the level of the master write post bus error interrupt.

SYSF LEVEL SYSFAIL Interrupt Level. These three bits define the level of the SYSFAIL interrupt.

Interrupt Level Register 1 (bits 8-15)

REG	Interrupt Level Register 1 (bits 8-15)							
ADR/SIZ	$FFF40078 (8 bits [6 used] of 32)							
BIT	15	14	13	12	11	10	9	8
FIELD		PE LEVEL				IRQ1E LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the VMEbus IRQ1 edge-sensitive interrupt and the level of the external (parity error) interrupt.

IRQ1E LEVEL IRQ1E Interrupt Level. These three bits define the level of the VMEbus IRQ1 edge-sensitive interrupt.

PE LEVEL Parity Error Interrupt Level. These three bits define the level of the external (parity error) interrupt.

Interrupt Level Register 1 (bits 0-7)

REG	Interrupt Level Register 1 (bits 0-7)							
ADR/SIZ	$FFF40078 (8 bits [6 used] of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD		TICK2 LEVEL				TICK1 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the tick timer 1 interrupt and the tick timer 2 interrupt.

TICK1 LEVEL Tick Timer 1 Interrupt Level. These three bits define the level of the tick timer 1 interrupt.

TICK2 LEVEL Tick Timer 2 Interrupt Level. These three bits define the level of the tick timer 2 interrupt.

Interrupt Level Register 2 (bits 24-31)

REG	Interrupt Level Register 2 (bits 24-31)							
ADR/SIZ	$FFF4007C (8 bits [6 used] of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD		VIA LEVEL				DMA LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the DMA controller interrupt and the VMEbus acknowledge interrupt.

DMA LEVEL DMA Interrupt Level. These three bits define the level of the DMA controller interrupt.

VIA LEVEL VMEbus Acknowledge Interrupt Level. These three bits define the level of the VMEbus interrupter acknowledge interrupt.

Interrupt Level Register 2 (bits 16-23)

REG	Interrupt Level Register 2 (bits 16-23)							
ADR/SIZ	$FFF4007C (8 bits [6 used] of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD		SIG3 LEVEL				SIG2 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the GCSR SIG2 interrupt and the GCSR SIG3 interrupt.

SIG2 LEVEL SIG2 Interrupt Level. These three bits define the level of the GCSR SIG2 interrupt.

SIG3 LEVEL SIG3 Interrupt Level. These three bits define the level of the GCSR SIG3 interrupt.

Interrupt Level Register 2 (bits 8-15)

REG	Interrupt Level Register 2 (bits 8-15)							
ADR/SIZ	$FFF4007C (8 bits [6 used] of 32)							
BIT	15	14	13	12	11	10	9	8
FIELD		SIG1 LEVEL				SIG0 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the GCSR SIG0 interrupt and the GCSR SIG1 interrupt.

SIG0 LEVEL SIG0 Interrupt Level. These three bits define the level of the GCSR SIG0 interrupt.

SIG1 LEVEL SIG1 Interrupt Level. These three bits define the level of the GCSR SIG1 interrupt.

Interrupt Level Register 2 (bits 0-7)

REG	Interrupt Level Register 2 (bits 0-7)							
ADR/SIZ	$FFF4007C (8 bits [6 used] of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD		LM1 LEVEL				LM0 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the GCSR LM0 interrupt and the GCSR LM1 interrupt.

LM0 LEVEL LM0 Interrupt Level. These three bits define the level of the GCSR LM0 interrupt.

LM1 LEVEL LM1 Interrupt Level. These three bits define the level of the GCSR LM1 interrupt.

Interrupt Level Register 3 (bits 24-31)

<table>
<tr><td>REG</td><td colspan="8">Interrupt Level Register 3 (bits 24-31)</td></tr>
<tr><td>ADR/SIZ</td><td colspan="8">$FFF40080 (8 bits [6 used] of 32)</td></tr>
<tr><td>BIT</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td></tr>
<tr><td>FIELD</td><td></td><td colspan="3">SW7 LEVEL</td><td></td><td colspan="3">SW6 LEVEL</td></tr>
<tr><td>OPER</td><td></td><td colspan="3">R/W</td><td></td><td colspan="3">R/W</td></tr>
<tr><td>RESET</td><td></td><td colspan="3">0 PSL</td><td></td><td colspan="3">0 PSL</td></tr>
</table>

This register is used to define the level of the software 6 interrupt and the software 7 interrupt.

SW6 LEVEL Software 6 Interrupt Level. These three bits define the level of the software 6 interrupt.

SW7 LEVEL Software 7 Interrupt Level. These three bits define the level of the software 7 interrupt.

Interrupt Level Register 3 (bits 16-23)

<table>
<tr><td>REG</td><td colspan="8">Interrupt Level Register 3 (bits 16-23)</td></tr>
<tr><td>ADR/SIZ</td><td colspan="8">$FFF40080 (8 bits [6 used] of 32)</td></tr>
<tr><td>BIT</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td>FIELD</td><td></td><td colspan="3">SIG5 LEVEL</td><td></td><td colspan="3">SIG4 LEVEL</td></tr>
<tr><td>OPER</td><td></td><td colspan="3">R/W</td><td></td><td colspan="3">R/W</td></tr>
<tr><td>RESET</td><td></td><td colspan="3">0 PSL</td><td></td><td colspan="3">0 PSL</td></tr>
</table>

This register is used to define the level of the software 4 interrupt and the software 5 interrupt.

SW4 LEVEL Software 4 Interrupt Level. These three bits define the level of the software 4 interrupt.

SW5 LEVEL Software 5 Interrupt Level. These three bits define the level of the software 5 interrupt.

Interrupt Level Register 3 (bits 8-15)

REG	Interrupt Level Register 3 (bits 8-15)							
ADR/SIZ	$FFF40080 (8 bits [6 used] of 32)							
BIT	15	14	13	12	11	10	9	8
FIELD		SW3 LEVEL				SW2 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the software 2 interrupt and the software 3 interrupt.

SW2 LEVEL Software 2 Interrupt Level. These three bits define the level of the software 2 interrupt.

SW3 LEVEL Software 3 Interrupt Level. These three bits define the level of the software 3 interrupt.

Interrupt Level Register 3 (bits 0-7)

REG	Interrupt Level Register 3 (bits 0-7)							
ADR/SIZ	$FFF40080 (8 bits [6 used] of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD		SW1 LEVEL				SW0 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the software 0 interrupt and the software 1 interrupt.

SW0 LEVEL Software 0 Interrupt Level. These three bits define the level of the software 0 interrupt.

SW1 LEVEL Software 1 Interrupt Level. These three bits define the level of the software 1 interrupt.

Interrupt Level Register 4 (bits 24-31)

REG	Interrupt Level Register 4 (bits 24-31)							
ADR/SIZ	$FFF40084 (8 bits [6 used] of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD		SPARE LEVEL				VIRQ7 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the VMEbus IRQ7 interrupt and the spare interrupt. The VMEbus level 7 (IRQ7) interrupt may be mapped to any local peripheral bus interrupt level.

VIRQ7 LEVEL VMEbus IRQ7 Interrupt Level. These three bits define the level of the VMEbus IRQ7 interrupt.

SPARE LEVEL Spare Interrupt Level. These three bits define the level of the spare interrupt.

Interrupt Level Register 4 (bits 16-23)

REG	Interrupt Level Register 4 (bits 16-23)							
ADR/SIZ	$FFF40084 (8 bits [6 used] of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD		VIRQ6 LEVEL				VIRQ5 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the VMEbus IRQ5 interrupt and the VMEbus IRQ6 interrupt. The VMEbus level 5 (IRQ5) interrupt and the VME bus level 6 (IRQ6) interrupt may be mapped to any local peripheral bus interrupt level.

VIRQ5 LEVEL VMEbus IRQ5 Interrupt Level. These three bits define the level of the VMEbus IRQ5 interrupt.

VIRQ6 LEVEL VMEbus IRQ5 Interrupt Level. These three bits define the level of the VMEbus IRQ6 interrupt.

Interrupt Level Register 4 (bits 8-15)

REG	Interrupt Level Register 4 (bits 8-15)							
ADR/SIZ	$FFF40084 (8 bits [6 used] of 32)							
BIT	15	14	13	12	11	10	9	8
FIELD		VIRQ4 LEVEL				VIRQ3 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the VMEbus IRQ3 interrupt and the VMEbus IRQ4 interrupt. The VMEbus level 3 (IRQ3) interrupt and the VMEbus level 4 (IRQ4) interrupt may be mapped to any local peripheral bus interrupt level.

VIRQ3 LEVEL VMEbus IRQ3 Interrupt Level. These three bits define the level of the VMEbus IRQ3 interrupt.

VIRQ4 LEVEL VMEbus IRQ4 Interrupt Level. These three bits define the level of the VMEbus IRQ4 interrupt.

Interrupt Level Register 4 (bits 0-7)

REG	Interrupt Level Register 4 (bits 0-7)							
ADR/SIZ	$FFF40084 (8 bits [6 used] of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD		VIRQ2 LEVEL				VIRQ1 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the VMEbus IRQ1 interrupt and the VMEbus IRQ2 interrupt. The VMEbus level 1 (IRQ1) interrupt and the VMEbus level 2 (IRQ2) interrupt may be mapped to any local peripheral bus interrupt level.

VIRQ1 LEVEL VMEbus IRQ1 Interrupt Level. These three bits define the level of the VMEbus IRQ1 interrupt.

VIRQ2 LEVEL VMEbus IRQ2 Interrupt Level. These three bits define the level of the VMEbus IRQ2 interrupt.

Vector Base Register

REG	Vector Base Register							
ADR/SIZ	$FFF40088 (8 bits of 32)							
BIT	31	30	29	28	27	26	25	24
FIELD	VBR 0				VBR 1			
OPER	R/W				R/W			
RESET	0 PSL				0 PSL			

This register is used to define the interrupt base vectors.

VBR 1 Vector Base 1. These four bits define the interrupt base vector 1.

VBR 0 Vector Base 1. These four bits define the interrupt base vector 0.

Notes

1. **Refer to Table 2-3, Local Peripheral Bus Interrupter Summary, earlier in this chapter, for further information.**
2. **A suggested setting for the Vector Base Register for the VMEchip2 is: VBR0 = 6, VBR1 = 7 (i.e., setting the Vector Base Register at address $FFF40088 to $67xxxxxx). This produces a Vector Base0 of $60 corresponding to the "X" in Table 2-3, and a Vector Base1 of $70 corresponding to the "Y" in Table 2-3.**

I/O Control Register 1

REG	I/O Control Register 1							
ADR/SIZ	$FFF40088 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
FIELD	MIEN	SYSFL	ACFL	ABRTL	GPIOEN3	GPIOEN2	GPIOEN1	GPIOEN0
OPER	R/W	R	R	R	R/W	R/W	R/W	R/W
RESET	0 PSL	X	X	X	0 PS	0 PS	0 PS	0 PS

This register is a general purpose I/O control register. Bits 16-19 control the direction of the four General Purpose I/O pins (GPIO0-3).

GPIOEN0 General Purpose I/O Enable 0. When this bit is low, the GPIO0 pin is an input. When this bit is high, the GPIO0 pin is an output.

GPIOEN1 General Purpose I/O Enable 1. When this bit is low, the GPIO1 pin is an input. When this bit is high, the GPIO1 pin is an output.

GPIOEN2 General Purpose I/O Enable 2. When this bit is low, the GPIO2 pin is an input. When this bit is high, the GPIO2 pin is an output.

GPIOEN3 General Purpose I/O Enable 3. When this bit is low, the GPIO3 pin is an input. When this bit is high, the GPIO3 pin is an output.

ABRTL Abort Level. This bit indicates the status of the ABORT switch. When this bit is high, the ABORT switch is depressed. When this bit is low, the ABORT switch is not depressed.

ACFL ACFAIL Level. This bit indicates the status of the ACFAIL signal line on the VMEbus. When this bit is high, the ACFAIL signal line is active. When this bit is low, the ACFAIL signal line is not active.

SYSFL SYSFAIL Level. This bit indicates the status of the SYSFAIL signal line on the VMEbus. When this bit is high, the SYSFAIL signal line is active. When this bit is low, the SYSFAIL signal line is not active.

MIEN Master Interrupt Enable. When this bit is low, all interrupts controlled by the VMEchip2 are masked. When this bit is high, all interrupts controlled by the VMEchip2 are not masked.

I/O Control Register 2

REG	I/O Control Register 2							
ADR/SIZ	$FFF40088 (8 bits of 32)							
BIT	15	14	13	12	11	10	0	8
FIELD	GPIOO3	GPIOO2	GPIOO1	GPIOO0	GPIOI3	GPIOI2	GPIOI1	GPIOI0
OPER	R/W	R/W	R/W	R/W	R	R	R	R
RESET	0 PSL	0 PS	0 PS	0 PS	X	X	X	X

This register is a general purpose I/O control register. Bits 8-11 reflect the status of the four General Purpose I/O pins (GPIO0-3).

GPIOI0 General Purpose I/O Input 0. When this bit is low, the GPIO0 pin is low. When this bit is high, the GPIO0 pin is high.

GPIOI1 General Purpose I/O Input 1. When this bit is low, the GPIO1 pin is low. When this bit is high, the GPIO1 pin is high.

GPIOI2 General Purpose I/O Input 2. When this bit is low, the GPIO2 pin is low. When this bit is high, the GPIO2 pin is high.

GPIOI3 General Purpose I/O Input 3. When this bit is low, the GPIO3 pin is low. When this bit is high, the GPIO3 pin is high.

Bits 12-15 determine the driven level of the four General Purpose I/O pins (GPIO0-3) when they are defined as outputs.

GPIOO0 General Purpose I/O Output 0. When this bit is low, the GPIO0 pin is driven low if it is defined as an output. When this bit is high, the GPIO0 pin is driven high if it is defined as an output.

GPIOO1 General Purpose I/O Output 1. When this bit is low, the GPIO1 pin is driven low if it is defined as an output. When this bit is high, the GPIO1 pin is driven high if it is defined as an output.

GPIOO2 General Purpose I/O Output 2. When this bit is low, the GPIO2 pin is driven low if it is defined as an output. When this bit is high, the GPIO2 pin is driven high if it is defined as an output.

GPIOO3 General Purpose I/O Output 3. When this bit is low, the GPIO3 pin is driven low if it is defined as an output. When this bit is high, the GPIO3 pin is driven high if it is defined as an output.

Notes

1. **The GPIO0 pin on the MVME197 is used to monitor the +12 Vdc power to the LAN connector. When the GPIOI0 bit is high, +12 Vdc is not present. When the GPIOI0 bit is low, +12 Vdc is present.**
2. **The GPIO1 pin on the MVME197 is used to control the STS LED* signal, which is at the remote RAL connector. When the GPIO1 pin is high or programmed as an input, the LED is on. When the GPIO1 pin is low, the LED is off.**
3. **The GPIO2 pin on the MVME197 is used to control bus error handling by the 82596CA interface logic. Refer to the *82596CA LAN Controller Interface* section in the *PCCchip2* chapter.**
4. **The GPIO1 and GPIO3 pins on the MVME197 are connected to the remote reset connector J1 pins 16 and 18, respectively.**
5. **On the MVME197, the GPIO3 pin may be used as an input or output.**

I/O Control Register 3

REG	I/O Control Register 3							
ADR/SIZ	$FFF40088 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
FIELD	GPI7	GPI6	GPI5	GPI4	GPI3	GPI2	GPI1	GPI0
OPER	R	R	R	R	R	R	R	R
RESET	X	X	X	X	X	X	X	X

This register reflects the status of the eight General Purpose Input pins (GPIO-GPIO7). On the MVME197, the GPI pins are connected to the general purpose switch S3. (Refer to the *Board Level Hardware Description* chapter for details on selected switch settings).

GPI0 General Purpose Input 0. When this bit is low, pin GPI0 is low. When this bit is high, pin GPI0 is high.

GPI1 General Purpose Input 1. When this bit is low, pin GPI1 is low. When this bit is high, pin GPI1 is high.

GPI2 General Purpose Input 2. When this bit is low, pin GPI2 is low. When this bit is high, pin GPI2 is high.

GPI3 General Purpose Input 3. When this bit is low, pin GPI3 is low. When this bit is high, pin GPI3 is high.

GPI4 General Purpose Input 4. When this bit is low, pin GPI4 is low. When this bit is high, pin GPI4 is high.

GPI5 General Purpose Input 5. When this bit is low, pin GPI5 is low. When this bit is high, pin GPI5 is high.

GPI6 General Purpose Input 6. When this bit is low, pin GPI6 is low. When this bit is high, pin GPI6 is high.

GPI7 General Purpose Input 7. When this bit is low, pin GPI7 is low. When this bit is high, pin GPI7 is high.

GCSR Programming Model

This section describes the programming model for the Global Status and Control Registers (GCSR) in the VMEchip2. The local peripheral bus map decoder for the GCSR registers is included in the VMEchip2. The local peripheral bus base address for the GCSR is $FFF40100. The registers in the GCSR are 16 bits wide and they are byte accessible from both the VMEbus and the local peripheral bus. The GCSR is located in the 16-bit VMEbus short I/O space and it responds to address modifier codes $29 or $2D. The address of the GCSR as viewed from the VMEbus depends upon the GCSR group select value XX and GCSR board select value Y programmed in the LCSR. The board value Y may be $0 through $E, allowing 15 boards in one group. The value $F is reserved for the location monitors.

The VMEchip2 includes four location monitors (LM0-LM3). The location monitors provide a broadcast signaling capability on the VMEbus. When a location monitor address is generated on the VMEbus, all location monitors in the group are cleared. The signal interrupts SIG0-SIG3 should be used to signal individual boards. The location monitors are located in the VMEbus short I/O space and the specific address is determined by the VMEchip2 group address. The location monitors LM0-LM3 are located at addresses $XXF1, $XXF3, $XXF5, and $XXF7, respectively. A location monitor cycle on the VMEbus is generated by a read or write to VMEbus short I/O address $XXFN, where XX is the group address and N is the specific location monitor address. When the VMEchip2 generates a location monitor cycle to the VMEbus, within its own group, the VMEchip2 DTACKs itself. A VMEchip2 cannot DTACK location monitor cycles to other groups.

The GCSR section of the VMEchip2 contains a chip ID register, a chip revision register, a location monitor status register, an interrupt control register, a board control register, and six general purpose registers. The chip ID and revision registers are provided to allow software to determine the ID of the chip and its revision level. The VMEchip2 has a chip ID of ten. ID codes zero and one are used by the old VMEchip. The initial revision of the VMEchip2 is zero. If mask changes are required, the revision level is incremented.

The location monitor status register provides the status of the location monitors. A location monitor bit is cleared when the VMEchip2 detects a VMEbus cycle to the corresponding location monitor address. When the LM0 or LM1 bits are cleared, an interrupt is set to the local peripheral bus interrupter. If the LM0 or LM1 interrupt is enabled in the local peripheral bus interrupter, then a local peripheral bus interrupt is generated. The location monitor bits are set by writing a one to the corresponding bit in the location monitor register. LM0 and LM1 can also be set by writing a one to the corresponding clear bits in the local interrupt clear register.

The interrupt control register provides four bits that allow the VMEbus to interrupt the local peripheral bus. An interrupt is sent to the local peripheral bus interrupter when one of the bits is set. If the interrupt is enabled in the local peripheral bus interrupter, then a local peripheral bus interrupt is generated. The interrupt bits are cleared by writing a one to the corresponding bit in the interrupt clear register.

The board control register allows a VMEbus master to reset the local peripheral bus, prevent the VMEchip2 from driving the SYSFAIL signal line, and detect if the VMEchip2 wants to drive the SYSFAIL signal line.

The six general purpose registers can be read and written from both the local peripheral bus and the VMEbus. These registers are provided to allow local peripheral bus masters to communicate with VMEbus masters. The function of these registers is not defined by this specification. The GCSR supports read-modify-write cycles such as TAS.

Note

The GCSR allows a VMEbus master to reset the local peripheral bus. This feature is very dangerous and should be used with caution. The local reset feature is a partial system reset, not a complete system reset such as power-up reset or SYSRESET. When the local peripheral bus reset signal is asserted, a local peripheral bus cycle may be aborted. The VMEchip2 is connected to both the local peripheral bus and the VMEbus and if the aborted cycle is bound for the VMEbus, erratic operation may result. Communications between the local processor and a VMEbus master should use interrupts or mailbox locations; reset should not be used in normal communications. Reset should be used only when the local processor is halted or the local peripheral bus is hung and reset is the last resort.

Programming the GCSR

A complete description of the GCSR is provided in the following tables. Each register definition includes a table with 5 lines. Line 1 is the base address of the register as viewed from the local peripheral bus and as viewed from the VMEbus, and the number of bits defined in the table. Line 2 shows the bits defined by this table. Line 3 defines the name of the register or the name of the bits in the register.

Line 4 defines the operations possible on the register bits as follows:

1. R - This bit is a read-only status bit.
2. R/W - This bit is readable and writable.
3. S/R - Writing a one to this bit sets it. Reading it returns its current status.

Line 5 defines the state of the bit following a reset as defined below.

1. P - The bit is affected by power-up reset.
2. S - The bit is affected by SYSRESET.
3. L - The bit is affected by local peripheral bus reset.
4. X - The bit is not affected by any reset.

A summary of the GCSR is shown in Table 2-4.

Table 2-1. VMEchip2 Memory Map - GCSR Summary

VMEchip2 GCSR Base Address = $FFF40100

L	V	15	Bit Numbers						8	7	Bit Numbers						0
0	0	CHIP REVISION								CHIP ID							
4	2	LM3	LM2	LM1	LM0	SIG3	SIG2	SIG1	SIG0	RST	ISF	BF	SCON	SYSFL	X	X	X
8	4	GENERAL PURPOSE CONTROL AND STATUS REGISTER 0															
C	6	GENERAL PURPOSE CONTROL AND STATUS REGISTER 1															
10	8	GENERAL PURPOSE CONTROL AND STATUS REGISTER 2															
14	A	GENERAL PURPOSE CONTROL AND STATUS REGISTER 3															
18	C	GENERAL PURPOSE CONTROL AND STATUS REGISTER 4															
1C	E	GENERAL PURPOSE CONTROL AND STATUS REGISTER 5															

Notes

L = Local (peripheral) bus offset.

V = VMEbus offset.

VMEchip2 Revision Register

REG	VMEchip2 Revision Register							
ADR/SIZ	Local peripheral bus: $FFF40100/VMEbus: $XXY0 (8 bits)							
BIT	15	14	13	12	11	10	0	8
FIELD	Chip Revision							
OPER	R							
RESET	0 PS							

This register is the VMEchip2 revision register. The revision level for the VMEchip2 starts at zero and is incremented if mask changes are required.

VMEchip2 ID Register

REG	VMEchip2 ID Register							
ADR/SIZ	Local peripheral bus: $FFF40100/VMEbus: $XXY0 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	Chip ID							
OPER	R							
RESET	10 PS							

This register is the VMEchip2 ID register. The ID for the VMEchip2 is 10.

VMEchip2 LM/SIGI Register

REG	VMEchip2 LM/SIGI Register							
ADR/SIZ	Local peripheral bus: $FFF40104/VMEbus: $XXY2 (8 bits)							
BIT	15	14	13	12	11	10	0	8
FIELD	LM3	LM2	LM1	LM0	SIG3	SIG2	SIG1	SIG0
OPER	R	R	R	R	S/R	S/R	S/R	S/R
RESET	1 PS	1 PS	1 PS	1 PS	0 PS	0 PS	0 PS	0 PS

This register is the VMEchip2 location monitor register and the interrupt register.

SIG0 Signal Interrupt 0. The SIG0 bit is set when a VMEbus master writes a one to it. When the SIG0 bit is set, an interrupt is sent to the local peripheral bus interrupter. The SIG0 bit is cleared when the local processor writes a one to the SIG0 bit in this register or the CSIG0 bit in the local interrupt clear register.

SIG1 Signal Interrupt 1. The SIG1 bit is set when a VMEbus master writes a one to it. When the SIG1 bit is set, an interrupt is sent to the local peripheral bus interrupter. The SIG1 bit is cleared when the local processor writes a one to the SIG1 bit in this register or the CSIG1 bit in the local interrupt clear register.

SIG2 Signal Interrupt 2. The SIG2 bit is set when a VMEbus master writes a one to it. When the SIG2 bit is set, an interrupt is sent to the local peripheral bus interrupter. The SIG2 bit is cleared when the local processor writes a one to the SIG2 bit in this register or the CSIG2 bit in the local interrupt clear register.

SIG3 Signal Interrupt 3. The SIG3 bit is set when a VMEbus master writes a one to it. When the SIG3 bit is set, an interrupt is sent to the local peripheral bus interrupter. The SIG3 bit is cleared when the local processor writes a one to the SIG3 bit in this register or the CSIG3 bit in the local interrupt clear register.

LM0 Location Monitor 0. This bit is cleared by an LM0 cycle on the VMEbus. When this bit is cleared, an interrupt is set to the local peripheral bus interrupter. This bit is set when the local processor or a VMEbus master writes a one to the LM0 bit in this register or the CLM0 bit in local interrupt clear register.

LM1 Location Monitor 1. This bit is cleared by an LM1 cycle on the VMEbus. When this bit is cleared, an interrupt is set to the local peripheral bus interrupter. This bit is set when the local processor or a VMEbus master writes a one to the LM1 bit in this register or the CLM1 bit in local interrupt clear register.

LM2 Location Monitor 2. This bit is cleared by an LM2 cycle on the VMEbus. This bit is set when the local processor or a VMEbus master writes a one to the LM0 bit in this register.

LM3 Location Monitor 3. This bit is cleared by an LM3 cycle on the VMEbus. This bit is set when the local processor or a VMEbus master writes a one to the LM3 bit in this register.

VMEchip2 Board Status/Control Register

REG	VMEchip2 Board Status/Control Register							
ADR/SIZ	Local peripheral bus: $FFF40104/VMEbus: $XXY2 (8 bits [5 used])							
BIT	7	6	5	4	3	2	1	0
FIELD	RST	ISF	BF	SCON	SYSFL			
OPER	S/R	R/W	R	R	R			
RESET	0 PSL	0 PSL	1 PS	X	1 PSL			

This register is the VMEchip2 board status/control register.

SYSFL SYSFAIL. This bit is set when the VMEchip2 is driving the SYSFAIL signal.

SCON System Controller. This bit is set if the VMEchip2 is system controller.

BF Board Fail. When this bit is high, the Board Fail signal is active. When this bit is low, the Board Fail signal is inactive. When this bit is set, the VMEchip2 drives SYSFAIL if the inhibit SYSFAIL bit is not set.

ISF When this bit is set, the VMEchip2 is prevented from driving the VMEbus SYSFAIL signal line. When this bit is cleared, the VMEchip2 is allowed to drive the VMEbus SYSFAIL signal line.

RST Reset. This bit allows a VMEbus master to reset and hold in reset the local peripheral bus. See the note on local reset in the *GCSR Programming Model* section, earlier in this chapter. When this bit is set, a local peripheral bus reset is generated. This bit is cleared by the local peripheral bus reset.

General Purpose Control and Status Register 0

REG	General Purpose Control and Status Register 0															
ADR/SIZ	Local peripheral bus: $FFF40108/VMEbus: $XXY4 (16 bits)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	General Purpose 0															
OPER	R/W															
RESET	0 PS															

This register is a general purpose register that allows a local peripheral bus master to communicate with a VMEbus master. The function of this register is not defined by the hardware specification.

General Purpose Control and Status Register 1

REG	General Purpose Control and Status Register 1															
ADR/SIZ	Local peripheral bus: $FFF4010C/VMEbus: $XXY6 (16 bits)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	General Purpose 1															
OPER	R/W															
RESET	0 PS															

This register is a general purpose register that allows a local peripheral bus master to communicate with a VMEbus master. The function of this register is not defined by the hardware specification.

General Purpose Control and Status Register 2

REG	General Purpose Control and Status Register 2															
ADR/SIZ	Local peripheral bus: $FFF40110/VMEbus: $XXY8 (16 bits)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	General Purpose 2															
OPER	R/W															
RESET	0 PS															

This register is a general purpose register that allows a local peripheral bus master to communicate with a VMEbus master. The function of this register is not defined by the hardware specification.

General Purpose Control and Status Register 3

REG	General Purpose Control and Status Register 3															
ADR/SIZ	Local peripheral bus: $FFF40114/VMEbus: $XXYA (16 bits)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	General Purpose 3															
OPER	R/W															
RESET	0 PS															

This register is a general purpose register that allows a local peripheral bus master to communicate with a VMEbus master. The function of this register is not defined by the hardware specification.

General Purpose Control and Status Register 4

REG	General Purpose Control and Status Register 4															
ADR/SIZ	Local peripheral bus: $FFF40118/VMEbus: $XXYC (16 bits)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	General Purpose 4															
OPER	R/W															
RESET	0 PS															

This register is a general purpose register that allows a local peripheral bus master to communicate with a VMEbus master. The function of this register is not defined by the hardware specification.

General Purpose Control and Status Register 5

REG	General Purpose Control and Status Register 5															
ADR/SIZ	Local peripheral bus: $FFF4011C/VMEbus: $XXYE (16 bits)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	General Purpose 5															
OPER	R/W															
RESET	0 PS															

This register is a general purpose register that allows a local peripheral bus master to communicate with a VMEbus master. The function of this register is not defined by the hardware specification.

PCCchip2 3

Introduction

This chapter defines the peripheral channel controller ASIC which is hereafter referred to as the PCCchip2. The PCCchip2 is designed for the MVME197 series of Single Board Computers to interface the local peripheral bus (MC68040 compatible bus) to various peripheral devices.

PCCchip2 Features

- ❑ BBRAM interface with dynamic sizing support
- ❑ DROM interface with dynamic sizing support
- ❑ 8-bit parallel I\O port
- ❑ Master and slave interface for CD2401 Intelligent Multi-Protocol Peripheral
- ❑ Host interface to Intel 82596CA LAN Coprocessor
- ❑ Host interface to NCR SCSI I\O Processor
- ❑ Two 32-bit tick timers
- ❑ Interrupt handler for tick timers and all peripherals:
  - All interrupts are level-programmable
  - All interrupts are maskable
  - All interrupts provide a unique vector
- ❑ Interrupt Mask Register to help prioritize Interrupt Requests to the MC88110 (This feature is not applicable to any of the MVME197 module series)

Functional Description

The following sections provide an overview of the functions provided by the PCCchip2. A detailed programming model for the PCCchip2 control and status registers is provided in a later section.

General Description

The PCCchip2 interfaces the local peripheral bus to certain peripherals on the MVME197 series of Single Board Computers including: the battery backup RAM (BBRAM), the download ROM (DROM), the Serial Communications

Controller (CL-CD2401), the LAN controller (82596CA), and the SCSI controller (NCR53C710). The PCCchip2 also provides two 32-bit timers and a parallel I/O port. The block diagram of the PCCchip2 is shown in Figure 3-1. On the MVME197, the MEMC040 functions are not utilized.

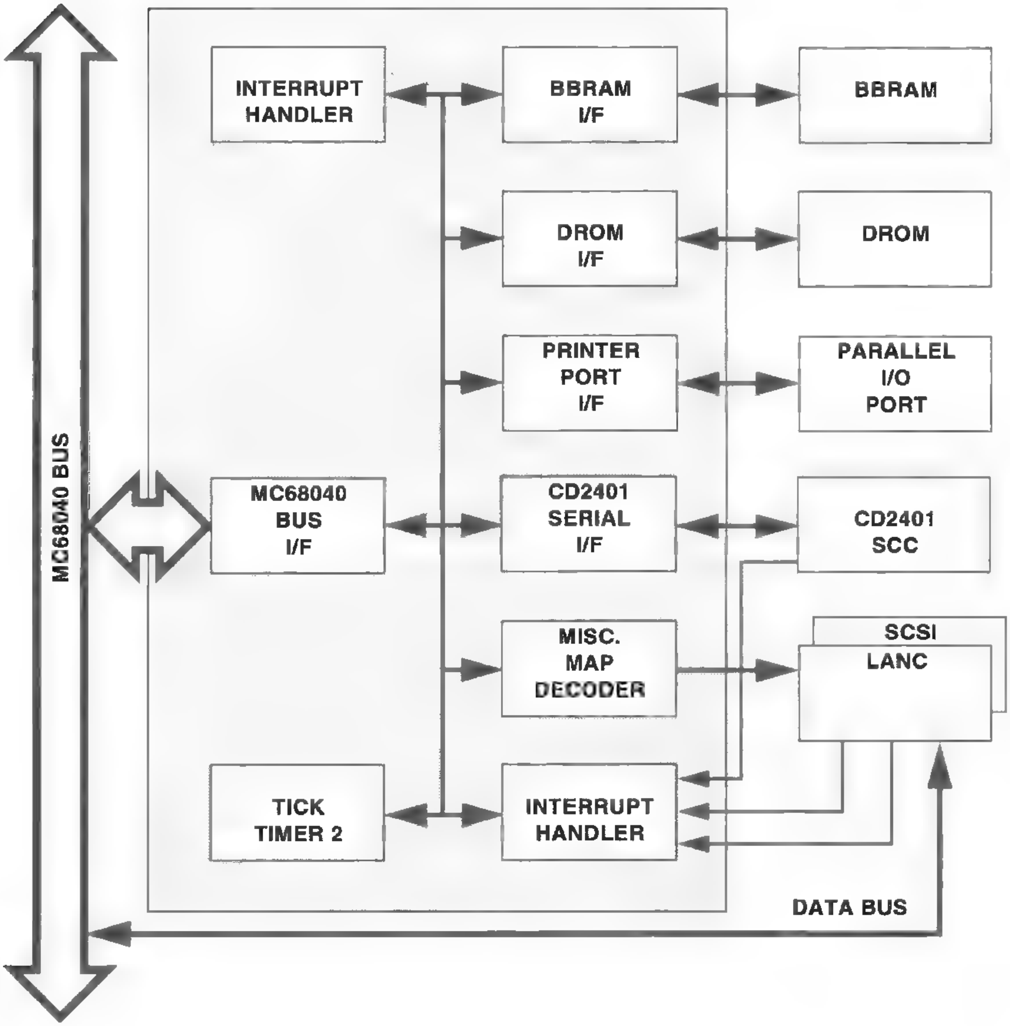


Figure 3-1. PCCchip2 Block Diagram

BBRAM Interface

The PCCchip2 provides a read/write interface to the BBRAM by any bus master on the MC68040 bus. The PCCchip2 performs dynamic sizing for accesses to the 8-bit BBRAM to make it appear contiguous. This feature allows code to be executable from the BBRAM. The BBRAM device access time must be no greater than 5 BCLK periods in fast mode or 9 BCLK periods in slow mode (notice that the BCLK is half the frequency of the processor clock). The BBRAM speed option is controlled by a control bit (FAST) in the General Control Register.

Download ROM Interface

The PCCchip2 provides a read/write interface to the download ROM (DROM) for any master on the MC68040 bus. The PCCchip2 performs dynamic sizing for accesses to the 8-bit DROM to make it appear contiguous. This feature allows code to be executable from the DROM. The DROM device access time must be no greater than 5 BCLK periods in fast mode or 9 BCLK periods in slow mode. The DROM speed option is controlled by a control bit (FAST) in the General Control Register.

If the DR0 bit is set in the General Control Register, DROM appears at locations $00000000 - $0001FFFF in addition to its normal address range.

DR0 is normally cleared at Local or Power-up Reset. However, if no other device responds to the first memory access on the local MC68040, after reset, the PCCchip2 sets DR0, causing DROM to respond to the memory access. DR0 will remain set until software writes a 0 to it. (The PCCchip2 determines that no device is responding to the vector fetch by detecting the lack of TA* or TEA* for 32 BCLK cycles after the assertion of TS*).

82596CA LAN Controller Interface

The LAN controller interface is described in the following sections.

MPU Port and MPU Channel Attention

The PCCchip2 allows the MC68040 bus master to communicate directly with the Intel 82596CA LAN Coprocessor by providing a map decoder and required control and timing logic. Two types of direct access are feasible with the 82596CA: MPU Port and MPU Attention.

MPU Port access enables the MC68040 bus master to write to an internal, 32-bit 82596CA command register. This allows it to do four things:

1. Write an alternate System Configuration Pointer address.
2. Write an alternative dump area pointer and perform a dump.
3. Execute a software reset.
4. Execute a self-test.

Each Port access must consist of two 16-bit writes: Upper Command Word (two bytes) and Lower Command Word (two bytes). The Upper Command Word (two bytes) is mapped at $FFF46000 and the Lower Command Word (two bytes) is mapped at $FFF46002.

The PCCchip2 only supports (decodes) MPU Port writes. It does not decode MPU Port reads. (Nor does the 82596CA support MPU Port reads.)

MPU Channel Attention access is used to cause the 82596CA to begin executing memory resident Command blocks. To execute a MPU Channel Attention, the MC68040 bus master performs a simple read or write to address $FFF46004.

MC68040 Bus Master Support for 82596CA

The 82596CA has DMA capability with an Intel i486-bus interface. When it is the local peripheral bus master, external hardware is needed to convert its bus cycles into MC68040 bus cycles. When the 82596CA has local peripheral bus mastership, the PCCchip2 will drive the following MC68040 signal lines:

❑ Snoop Control SC1-SC0 (with the value programmed into the LAN interrupt control register) (This Snoop Control feature is not applicable to any of the MVME197 module series)

❑ Transfer Types TT1-TT0 (with the value of %00)

❑ Transfer Modifiers TM2-TM0 (with the value of %101)

❑ Transfer Acknowledge (TA*) if Transfer Error Acknowledge (TEA*) is detected

LANC Bus Error

The 82596CA does not provide a way to terminate a bus cycle with an error indication. The interface to the 82596CA on the MVME197 provides several ways of processing bus errors that occur while the 82596CA is local peripheral bus master. These options are controlled by registers in the VMEchip2 and the PCCchip2.

The GPIO2 signal on the VMEchip2 LCSR (address $FFF40088) controls how the 82596CA interface logic responds to bus errors. If the GPIO2 signal is programmed as an input (reset state) or programmed as an output and set high, bus errors are processed in the following way. (NOTE: this option is not supported on Rev. A and Rev. B artwork boards.)

The 82596CA interface logic monitors all bus cycles initiated by the 82596CA, and if a bus error is indicated (TEA* = 0 and TA* = 1), the Back Off signal (BOFF*) to the 82596CA is asserted to keep the 82596CA off the local peripheral bus and prevent it from transmitting bad data or corrupting local memory. The LANC Error Status Register in the PCCchip2 is updated and a LANC bus error interrupt is generated if it is enabled in the PCCchip2. The Back Off signal remains asserted until the 82596CA is reset via a port reset command. After the 82596CA is reset, pending operations must be restarted.

If the GPIO2 signal is programmed as an output and set low, bus errors are processed in the following way. The 82596CA interface logic monitors all bus cycles initiated by the 82596CA, and if a bus error is indicated (TEA* = 0 and TA* = 1), the interface logic asserts the TA* signal to terminate the bus cycle. The LANC Error Status Register in the PCCchip2 is updated and LANC bus

error interrupt is generated if it is enabled in the PCCchip2. In this case the 82596CA continues to operate and because the cycle was terminated with an error, the 82596CA may transmit bad data or corrupt memory.

LANC Interrupt

When the PCCchip2 detects a high level on the INT signal from the 82596CA, if such interrupts are enabled, it generates an interrupt to the local peripheral bus.

If the C040 bit is set, the interrupt request goes to the local peripheral bus via the EIPL* pins at the level that is programmed for LANC interrupts in the LANC Interrupt Control Register. The MVME197LE module uses this option.

If the C040 bit is cleared, the interrupt goes out via the INT pin (if the level that is programmed for LANC interrupts in the LANC Interrupt Control Register is higher than the level set in the Interrupt Mask Level Register).

When the MPU acknowledges the LANC interrupt, the PCCchip2 responds with the vector that corresponds to LANC interrupts.

53C710 SCSI Controller Interface

The PCCchip2 provides a map decoder and an interrupt handler for the NCR-53C710 SCSI I/O Processor. The base address for the 53C710 is $FFF47000.

When the PCCchip2 detects low a level on the IRQ* line from the 53C710, if such interrupts are enabled, it generates an interrupt to the MPU.

If the C040 bit is set, the interrupt request goes to the local peripheral bus via the EIPL* pins at the level that is programmed for SCSI interrupts in the SCSI Interrupt Control Register. The MVME197LE module uses this option.

If the C040 bit is cleared, the interrupt goes out via the INT pin (if the level that is programmed for SCSI interrupts in the SCSI Interrupt Control Register is higher than the level set in the Interrupt Mask Level Register).

Memory Controller MEMC040 Interface

This function is not used on the MVME197 series of single board computers.

Parallel Port Interface

The PCCchip2 provides an 8/16-bit bidirectional parallel port. All eight/ sixteen bits of the port must be either inputs or outputs (no individual selection). In addition to the 8/16 bits of data, there are two control pins and five status pins. Each of the status pins can generate an interrupt to the MPU in any of the following programmable conditions: high level, low level, high-to-low transition, or low-to-high transition. This port may be used as a parallel printer port or as a general parallel I/O port.

When used as a parallel printer port, the five status pins function as: Printer Acknowledge (ACK), Printer Fault (FAULT*), Printer Busy (BSY), Printer Select (SELECT), and Printer Paper Error (PE); while the control pins act as Printer Strobe (STROBE*), and Input Prime (INP*).

The PCCchip2 provides an auto-strobe feature similar to that of the MVME147 PCC. In auto-strobe mode, after a write to the printer data register, the PCCchip2 automatically asserts the STROBE* pin for a selected time specified by the Printer Fast Strobe control bit. In manual mode, the Printer Strobe control bit directly controls the state of the STROBE* pin.

General Purpose I/O Pin

The General Purpose I/O pin can be used as an input pin, as an output pin, or as both. The PCCchip2 has a status bit that reflects the state of the pin. The PCCchip2 also has a control bit that allows it to drive the pin, and another control bit that controls the level that is driven.

The input can be configured to generate an interrupt to the MPU in any of the following programmable conditions: high level, low level, high-to-low transition, or low-to-high transition.

CD2401 SCC Interface

The PCCchip2 provides the required logic to interface the CL-CD2401 (SCC) Intelligent Multi-Protocol Peripheral to the Local Peripheral Bus. The interface logic consists of a local master interface, a local slave interface, a CD2401 Host interface, a CD2401 DMA interface, a CD2401 interrupt handler, and a local peripheral bus requester.

The base address for the CL-CD2401 is $FFF45000. It has 8- and 16-bit registers only. Consequently it does not respond when accessed with a size of 4 bytes (SIZ1,0 = %00) or with a size of 16 bytes (SIZ1,0 = %11).

There are three interrupts sources from the SCC: the receive interrupt, the transmit interrupt, and the modem interrupt. The PCCchip2 provides the ability to individually program the priority level of each of these interrupt sources.

When the C040 bit is set, these interrupts are sent to the MPU via the EIPL* pins (at the programmed level). This is the option used on the MVME197LE module.

When the C040 bit is cleared, they are sent to the CPU via the INT pin. (The INT pin is only asserted if the programmed level of the interrupt source is higher than the level programmed into the Interrupt Priority Mask Level Register).

There are two interrupt acknowledge modes supported by the PCCchip2 for the SCC: auto vector and direct. In the auto vector mode, the PCCchip2 supplies the interrupt vector to the MPU. (No interrupt acknowledge cycle is seen by the CD2401). In the direct mode, the SCC supplies the vector to the MPU. (The PCCchip2 passes the interrupt acknowledge cycle on through to the CD2401. Note that the PCCchip2 drives the CD2401 A7-A0 pins with $01 for modem interrupt acknowledges, $02 for transmit interrupt acknowledges and $03 for receive interrupt acknowledges). The use of the auto vector mode is not recommended because the CD2401 can supply the vector and the CD2401 requires an interrupt acknowledge cycle.

In order to support polling with the CD2401, the PCCchip2 supports pseudo interrupt acknowledge (PIACK) cycles to the CD2401. (This is required since the CD2401 has no other way of clearing its interrupt requests). PIACK cycles happen as follows:

1. The local peripheral bus master waits for an IRQ bit to be set in one of the three SCC interrupt control registers.

2. The local peripheral bus master starts a normal read cycle to one of the three PIACK registers in the PCCchip2. (The three PIACK registers correspond to modem, transmit, and receive interrupts respectively).
3. The PCCchip2 upon detecting the start of the read, performs an interrupt acknowledge cycle to the CD2401. (The PCCchip2 drives the CD2401 A7 through A0 pins with a value that corresponds to the PIACK register that is being read. If the Modem PIACK register is being read, then A7 through A0 = $01. If the Transmit PIACK register is being read, then A7 through A0 = $02. If the Receive PIACK register is being read, then A7 through A0 = $03).
4. As the interrupt acknowledge cycle completes, the PCCchip2 places the vector being driven by the CD2401 onto the local peripheral bus D0 through D8 and D16 through D23 signals. (From the MPU point of view, the status read from the selected PCCchip2 PIACK register is the vector from the CD2401).
5. The PCCchip signals to the local peripheral bus (via TA*) that the read cycle is complete.

Interrupt Prioritizer

The PCCchip2 provides circuitry to support a seven level, priority, interrupt scheme, when the local MPU is an MC88110. This support circuit is enabled/disabled by the C040 bit in the General Control Register.

On the MVME197, the C040 bit should be set. The PCCchip2 then drives the level of its highest priority internal interrupt request onto the EIPL<2..0>* pins. It is intended that the EIPL pins be combined outside the chip with any VMEchip2 IPL lines. The BusSwitch does the combination work. The priority interrupt scheme is assumed to be provided in the MC68040 when the C040 bit is set.

Tick Timer

The PCCchip2 includes two 32-bit general purpose tick timers. The tick timers run on a 1MHz clock which is derived from the processor clock by a prescaler.

Each tick timer has a 32-bit counter, a 32-bit compare register, and a clear-on-compare enable bit. The counter is readable and writable at any time. These timers can be used to generate interrupts at various rates or the counters can be read at various times for interval timing. There are two modes of operation for these timers: free-running and clear-on-compare.

In free-running mode, the timers have a resolution of 1 μsec and roll over after the count reaches the maximum value $FFFFFFFF. The rollover period for the timers is 71.6 minutes.

When the counter is enabled in the clear-on-compare mode, it increments every 1 μsec until the counter value matches the value in the compare register. When a match occurs, the counter is cleared.

When a match occurs, in either mode, an interrupt is sent to the local peripheral bus interrupter and the overflow counter is incremented. An interrupt to the local peripheral bus is only generated if the tick timer interrupt is enabled by the local peripheral bus interrupter. The overflow counter can be cleared by writing a one to the overflow clear bit.

Overall Memory Map

The following memory map includes all devices selected by the PCCchip2 map decoders, including those internal to the chip and those external. These devices respond only when the Transfer Type signals carry the values of %00 or %01 which correspond to Normal and MOVE16 accesses on the MC68040 bus.

Address Range	Selected Device	Comments
$FFF42000 - $FFF4203F	PCCchip2 Registers	See Programming Model
$FFF42040 - $FFF42FFF	PCCchip2 Registers	Repeated
$FFF43000 - $FFF43FFF	Memory Controller	External Device
$FFF45000 - $FFF450FF	SCC	External Device
$FFF45100 - $FFF45FFF	SCC	Repeated
$FFF46000 - $FFF46FFF	LANC	External Device
$FFF47000 - $FFF47FFF	SCSI	External Device
$FFF80000 - $FFF9FFFF	DROM (BOOT ROM)	External Device
$FFFC0000 - $FFFCFFFF	BBRAM	External Device

CSR Programming Model

This section defines the programming model for the control and status registers (CSR) in the PCCchip2. The base address of the CSR is $FFF42000. The PCCchip2 control and status registers can be accessed as bytes (8 bits), half words (16 bits), or words (32 bits). The possible operations for each bit in the CSR are as follows:

R This bit is a read only status bit.

R/W This bit is readable and writable.

W/AC This bit can be set and it is automatically cleared. This bit can also be read.

C Writing a one to this bit clears this bit or another bit. This bit reads zero.

S Writing a one to this bit sets this bit or another bit. This bit reads zero.

0 This bit is read only. It always reads as 0.

The possible states of the bits after local and power-up reset are as defined below.

P The bit is affected by power-up reset.

L The bit is affected by local reset.

X The bit is not affected by reset.

V The effect of reset on this bit is variable.

0 The bit is always 0.

1 The bit is always 1.

A summary of the PCCchip2 CSR is shown in Table 3-1.

Table 3-1. PCCchip2 Memory Map - Control and Status Registers

PCCchip2 LCSR Base Address = $FFF42000

OFFSET:

	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
00	CHIP ID								CHIP REVISION								DR0					CPU 040	MSTR INT EN	FAST BRAM	VECTOR BASE							
04	TIC TIMER 1 COMPARE																															
08	TIC TIMER 1 COUNTER																															
0C	TIC TIMER 2 COMPARE																															
10	TIC TIMER 2 COUNTER																															
14	PRESCALER COUNT								PRESCALER CLOCK ADJUST								OVERFLOW COUNTER 2					CLR OVF 2	COC 2	TIC 2	OVERFLOW COUNTER 1					CLR OVF 1	COC 1	TIC 1
18	GPI PLTY	GPI E/L*	GPI INT	GPI IEN	GPI ICLR	GP IRQ LEVEL								GPI	GIOE	GPO			TIC2 INT	TIC2 IEN	TIC2 ICLR	TIC TIMER 2 IRQ LEVEL					TIC1 INT	TIC1 IEN	TIC1 ICLR	TIC TIMER 1 IRQ LEVEL		
1C				SCC RTRY ERR	SCC PAR ERR	SCC EXT ERR	SCC LTO ERR	SCC SCLR			SCC MDM	SCC MDM	SCC MDM	SCC MODEM IRQ LEVEL					SCC TX IRQ	SCC TX IEN	SCC TX AVEC	SCC TRANSMIT IRQ LEVEL			SCC SC1	SCC SC2	SCC IRQ	SCC IEN	SCC AVEC	SCC RECEIVE IRQ LEVEL		
20																									SCC MODEM PIACK							
24									SCC TRANSMIT PIACK																SCC RECEIVE PIACK							
28					LAN PAR ERR	LAN EXT ERR	LAN LTO ERR	LAN SCLR									LAN PLTY	LAN E/L*	LAN INT	LAN IEN	LAN ICLR	LAN IRQ LEVEL			LAN SC1	LAN SC2	LAN ERR INT	LAN ERR IEN	LAN ERR ICLR	LAN ERR IRQ LEVEL		
2C					SCSI PAR	SCSI EXT ERR	SCSI LTO ERR	SCSI SCLR																			SCSI IRQ	SCSI IEN		SCSI INT IRQ LEVEL		
30	PRTR ACK PLTY	PRTR ACK E/L*	PRTR ACK INT	PRTR ACK IEN	PRTR ACK ICLR	PRTR ACK IRQ LEVEL			PRTR FLT PLTY	PRTR FLT E/L*	PRTR FLT INT	PRTR FLT IEN	PRTR FLT ICLR	PRTR FAULT IRQ LEVEL			PRTR SEL PLTY	PRTR SEL E/L*	PRTR SEL INT	PRTR SEL IEN	PRTR SEL ICLR	PRTR SEL IRQ LEVEL			PRTR PE PLTY	PRTR PE E/L*	PRTR PE INT	PRTR PE EN	PRTR PE ICLR	PRTR PE IRQ LEVEL		
34	PRTR BSY PLTY	PRTR BSY E/L*	PRTR BSY INT	PRTR BSY IEN	PRTR BSY ICLR	PRTR BSY IRQ LEVEL											PRTR ANY INT			PRTR ACK	PRTR FLT	PRTR SEL	PRTR PE	PRTR BSY				PRTR DAT ENBL	PRTR INP	PRTR STB	PRTR FAST ASTB	PRTR MAN STB
38	CHIP SPEED																PRINTER DATA															
3C																						INTERRUPT IPL LEVEL								INTERRUPT MASK LEVEL		
	D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

SCC PROVIDES ITS OWN VECTORS

Chip ID Register

REG	Chip ID Register							
ADR/SIZ	$FFF42000 (8 bits)							
BIT	31	30	29	28	27	26	25	24
FIELD	CID7	CID6	CID5	CID4	CID3	CID2	CID1	CID0
OPER	R	R	R	R	R	R	R	R
RESET	0	0	1	0	0	0	0	0

The Chip ID Register is located at $FFF42000. It is an 8-bit read-only register that is hard-wired to a hexadecimal value of $20. Writes to this register are ignored; however, the PCCchip2 always terminates the cycles properly with TA*.

Chip Revision Register

REG	Chip Revision Register							
ADR/SIZ	$FFF42001 (8 bits)							
BIT	23	22	21	20	19	18	17	16
FIELD	REV7	REV6	REV5	REV4	REV3	REV2	REV1	REV0
OPER	R	R	R	R	R	R	R	R
RESET	0	0	0	0	0	0	0	0

The Chip Revision Register is located at $FFF42001. It is an 8-bit read-only register that is hard-wired to reflect the revision level of the PCCchip2 ASIC. The current value of this register is $00. Writes to this register are ignored; however, the PCCchip2 always terminates the cycles properly with TA*.

General Control Register

REG	General Control Register							
ADR/SIZ	$FFF42002 (8 bits)							
BIT	15	14	13	12	11	10	0	8
FIELD	DR0					C040	MIEN	FAST
OPER	R/W					R/W	R/W	R/W
RESET	V PL					0 P	0 PL	0 P

The General Control Register is located at $FFF42002. It is an 8-bit register that controls chip general functions.

Note **V=1 if no other device responds to the first memory access after Power-up or Local Reset. Otherwise V=0.**

FAST Fast BRAM/DROM. This control bit tailors the control circuit for BBRAM/DROM to the speed of BBRAM and DROM (BOOT ROM).

When operating at 25 MHz, the FAST bit should be cleared for devices with access times longer than 200 nsec (5 BCLK cycles; note that BCLK is half of the MC88110 frequency). The bit can be set for devices that have access times of 200 nsec or faster. It is not allowed to use devices slower than 360 nsec (9 BCLK cycles), at 25 MHz.

When operating at 33 MHz, the FAST bit should be cleared for devices with access times longer than 150 nsec (5 BCLK cycles). The bit can be set for devices that have access times 150 nsec or faster. It is not allowed to use devices slower than 270 nsec (9 BCLK cycles), at 33 Mhz.

MIEN Master Interrupt Enable. When this bit is high, interrupts from and via the PCCchip2 are allowed to reach the BusSwitch. When it is low, all interrupts from the PCCchip2 are disabled (this includes both the EIPL* pins and the INT pin). Also, when the bit is low, all interrupt acknowledge cycles to the PCC2 are passed on, via the IACKOUT* pin. This bit is cleared by a reset.

C040 CPU MC68040. This bit should be set when the local peripheral bus master is an MC68040 type bus master. It should be set for the MVME197 module series. When the bit is set, EIPL<2..0>* are driven as outputs which carry the priority encoded interrupt request from the PCCchip2 interrupt sources. When the bit is cleared, EIPL<2..0>* are not driven as outputs, but are inputs only.

DR0 Download ROM at 0. When this bit is cleared, DROM (BOOT ROM) appears only in its normal address range. When DR0 is set, DROM also appears at location $00000000 - $0001FFFF. DR0 is cleared by Power-up or Local Reset, but if no other device responds (within a certain amount of time) to the first memory access after the reset, then the PCCchip2 sets DR0. This causes the DROM to respond to the memory access (and all memory accesses thereafter until software clears DR0).

Vector Base Register

REG	Vector Base Register							
ADR/SIZ	$FFF42003 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0
OPER	R/W	R/W	R/W	R/W	R	R	R	R
RESET	0 PL	0 PL	0 PL	0 PL	1 PL	1 PL	1 PL	1 PL

The Interrupt Vector Base Register is located at $FFF42003. It is an 8-bit read/write register that is used to supply the vector to the interrupt handler during an interrupt acknowledge cycle for: the two internal tick timers, LAN interrupt, LAN BERR interrupt, SCSI interrupt, GPIO interrupt, and parallel port interrupts. Only the most significant four bits are used. The least significant four bits encode the interrupt source during the acknowledge cycle. The exception to this is that after reset occurs, the interrupt vector passed is $0F, which remains in effect until a write is generated to the Vector Base Register.

A normal read access to the Vector Base Register yields the value $0F if the read happens before it has been initialized. A normal read access yields all zeros on bits 0-3 and the value that was last written on bits 4-7 if the read happens after the Vector Base Register has been initialized.

The encoding for the interrupt sources is shown below, where IV3-IV0 refer to bits 3-0 of the vector passed during the IACK cycle:

Interrupt Source	IV3-IV0	Priority
Printer Port-BSY	$0	Lowest
Printer Port-PE	$1	↑
Printer Port-SELECT	$2	
Printer Port-FAULT	$3	
Printer Port-ACK	$4	
SCSI IRQ	$5	
LANC ERR	$6	
LANC IRQ	$7	
Tick Timer 2 IRQ	$8	
Tick Timer 1 IRQ	$9	
GPIO IRQ	$A	
Serial Modem IRQ (auto vector mode only)	$B	
Serial RX IRQ (auto vector mode only)	$C	↓
Serial TX IRQ (auto vector mode only)	$D	Highest

Programming the Tick Timers

This section provides addresses and bit level descriptions of the prescaler, tick timers, and various other timer registers.

Tick Timer 1 Compare Register

REG	Tick Timer 1 Compare Register					
ADR/SIZ	$FFF42004 (32 bits)					
BIT	31	------------------	16	15	------------------	0
FIELD	Tick Timer 1 Compare					
OPER	R/W					
RESET	0 P					

The Tick Timer 1 Compare Register is a 32-bit register located at $FFF42004. The count value of Tick Timer 1 is compared to this register. When they are equal, an interrupt is sent to the local peripheral bus interrupter and the overflow counter is incremented. If the clear-on-compare mode is enabled, the counter is also cleared. For periodic interrupts, the following equation should be used to determine the compare register value for a specific period.

$$\text{Compare Register Value} = T\ (\mu\text{sec})$$

When programming the tick timer for periodic interrupts, the counter should be cleared to zero by software and then enabled. If the counter does not initially start at zero, the time to the first interrupt may be longer or shorter than expected. The rollover time for the counter is 71.6 minutes.

Tick Timer 1 Counter Register

REG	Tick Timer 1 Counter Register					
ADR/SIZ	$FFF42008 (32 bits)					
BIT	31	------------------	16	15	------------------	0
FIELD	Tick Timer 1 Counter					
OPER	R/W					
RESET	X					

The Tick Timer 1 Counter is a 32-bit read/write register located at address $FFFF2008. When enabled, it increments every microsecond. Software may read or write the counter at any time.

Tick Timer 2 Compare Register

REG	Tick Timer 2 Compare Register					
ADR/SIZ	$FFF4200C (32 bits)					
BIT	31	------------------	16	15	------------------	0
FIELD	Tick Timer 2 Compare					
OPER	R/W					
RESET	0 P					

The Tick Timer 2 Compare Register is a 32-bit register located at $FFF4200C. The count value of Tick Timer 2 is compared to this register. When they are equal, an interrupt is sent to the local peripheral bus interrupter and the overflow counter is incremented. If the clear-on-compare mode is enabled, the counter is also cleared. For periodic interrupts, the following equation should be used to determine the compare register value for a specific period.

$$\text{Compare Register Value} = T\ (\mu\text{sec})$$

When programming the tick timer for periodic interrupts, the counter should be cleared to zero by software and then enabled. If the counter does not initially start at zero, the time to the first interrupt may be longer or shorter than expected. The rollover time for the counter is 71.6 minutes.

Tick Timer 2 Counter Register

REG	Tick Timer 2 Counter Register					
ADR/SIZ	$FFF42010 (32 bits)					
BIT	31	------------------	16	15	------------------	0
FIELD	Tick Timer 2 Counter					
OPER	R/W					
RESET	X					

The Tick Timer 2 Counter is a 32-bit read/write register located at address $FFFF2010. When enabled, it increments every microsecond. Software may read or write the counter at any time.

Prescaler Count Register

REG	Prescaler Count Register							
ADR/SIZ	$FFF42014 (8 bits)							
BIT	31	30	29	28	27	26	25	24
FIELD	Prescaler Count							
OPER	R/W							
RESET	X							

The Prescaler Count Register is an 8-bit counter used to generate the 1 MHz clock for the two tick timers. This register is a read-only register located at address $FFF42014. It increments to $FF at the BCLK frequency, then it is loaded from the Prescaler Clock Adjust Register. Note that BCLK is half the frequency of the MC88110 processor clock.

Prescaler Clock Adjust Register

REG	Prescaler Clock Adjust Register							
ADR/SIZ	$FFF42015 (8 bits)							
BIT	23	22	21	20	19	18	17	16
FIELD	Prescaler Clock Adjust							
OPER	R/W							
RESET	$DF P							

The Prescaler Clock Adjust Register is an 8-bit read/write register located at address $FFF42015. It is required to adjust the prescaler so that it maintains a 1 MHz clock source for the tick timers, regardless of what frequency is used for BCLK. To provide a 1 MHz clock to the tick timers, the prescaler adjust register should be programmed based on the following equation:

$$\text{Prescaler Clock Adjust Register} = 256 - \text{BCLK (MHz)}$$

For example, for operation at 20 MHz (i.e., a 40MHz board) the prescaler value is $EC, at 25 MHz (i.e., a 50MHz board) it is $E7, and at 33 MHz it is $DF.

Non-integer bus clocks introduce an error into the specified times for the tick timers. The tick timer clock can be derived by the following equation.

$$\text{Tick timer clk} = \text{BCLK} / (256 - \text{Prescaler Value})$$

The maximum clock frequency for the tick timers is the BCLK frequency divided by two. The value 255 ($FF) is not allowed to be programmed into this register. If a write with the value of $FF occurs to this register, the PCCchip2 terminates the cycle properly with TA*, but the register remains unchanged.

Tick Timer 2 Control Register

REG	Tick Timer 2 Control Register							
ADR/SIZ	$FFF42016 (8 bits)							
BIT	15	14	13	12	11	10	0	8
FIELD	OVF3	OVF2	OVF1	OVF0		COVF	COC	CEN
OPER	R	R	R	R		C	R/W	R/W
RESET	0 PL	0 PL	0 PL	0 PL		0 PL	0 PL	0 PL

This is an 8-bit read/write register that controls Tick Timer 2. It is located at address $FFF42016.

CEN Counter Enable. When this bit is high, the counter increments. When this bit is low, the counter does not increment.

COC Clear On Compare. When this bit is high, the counter is reset to zero when it compares with the compare register. When this bit is low, the counter is not reset.

COVF Clear Overflow Counter. The overflow counter is cleared when a one is written to this bit.

OVF3-OVF0 Overflow Counter (bits 3-0). These four bits are the outputs of the overflow counter. The overflow counter is incremented each time the tick timer sends an interrupt to the local peripheral bus interrupter. The overflow counter can be cleared by writing a one to the COVF control bit.

Tick Timer 1 Control Register

REG	Tick Timer 1 Control Register							
ADR/SIZ	$FFF42017 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	OVF3	OVF2	OVF1	OVF0		COVF	COC	CEN
OPER	R	R	R	R		C	R/W	R/W
RESET	0 PL	0 PL	0 PL	0 PL		0 PL	0 PL	0 PL

This is an 8-bit read/write register that controls Tick Timer 1. It is located at address $FFF42017.

CEN Counter Enable. When this bit is high, the counter increments. When this bit is low, the counter does not increment.

COC Clear On Compare. When this bit is high, the counter is reset to zero when it compares with the compare register. When this bit is low, the counter is not reset.

COVF Clear Overflow Counter. The overflow counter is cleared when a one is written to this bit.

OVF3-OVF0 Overflow Counter (bits 3-0). These four bits are the outputs of the overflow counter. The overflow counter is incremented each time the tick timer sends an interrupt to the local peripheral bus interrupter. The overflow counter can be cleared by writing a one to the COVF control bit.

General Purpose Input Interrupt Control Register

REG	General Purpose Input Interrupt Control Register							
ADR/SIZ	$FFF42018 (8 bits)							
BIT	31	30	29	28	27	26	25	24
FIELD	PLTY	E/L*	INT	IEN	ICLR	IL2	IL1	IL0
OPER	R/W	R/W	R	R/W	C	R/W	R/W	R/W
RESET	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL

This is an 8-bit read/write register located at $FFF42018. It controls general purpose interrupt functions.

IL2-IL0 Interrupt Level (2-0). These three bits select the interrupt level for the general purpose input/output (GPIO) pin. Level 0 does not generate an interrupt.

ICLR Interrupt Clear. In edge-sensitive mode, writing a logic 1 to this bit clears the INT status bit. This bit has no function in level-sensitive mode. This bit is always read as zero.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

INT Interrupt Status. When this bit is high, a general purpose input interrupt is being generated at the level programmed in IL2-IL0 (if nonzero).

E/L* Edge/Level Mode. When this bit is high, the interrupt is edge-sensitive. The interrupt is level-sensitive when this bit is low.

PLTY Polarity Mode. When this bit is low, interrupt is activated by either a rising edge on the GPIO pin or a high level on the GPIO pin (depending on the E/L* bit). When this bit is high, interrupt is activated by either a falling edge on the GPIO pin or a low level of the GPIO pin (depending on the E/L* bit). Note that if this bit is changed while the E/L* bit is set (or is being set), a GPIO interrupt may be generated. This can be avoided by setting the ICLR bit during write cycles that change the E/L* bit.

General Purpose Input/Output Pin Control Register

REG	General Purpose Input/Output Pin Control Register							
ADR/SIZ	$FFF42019 (8 bits)							
BIT	23	22	21	20	19	18	17	16
FIELD						GPI	GPOE	GPO
OPER						R	R/W	R/W
RESET						X	0 PL	0 PL

The General Purpose Input/Output Pin Control Register is located at $FFF42019. This 8-bit register (3 bits used) controls the following GPIO pin functions.

GPO When GPO is set, and GPOE is set, the GPIO pin is at a logic high level. When GPO is cleared, and GPOE is set, the GPIO pin is at a logic low level.

GPOE General Purpose Enable. This bit controls whether or not the PCCchip2 drives the GPIO pin. When GPOE is set, the PCCchip2 drives the GPIO pin. When GPOE is cleared, the PCCchip2 does not drive the GPIO pin.

GPI This bit reflects the state of the GPIO pin. It is set when GPIO is high and cleared when GPIO is low.

Tick Timer 2 Interrupt Control Register

REG	Tick Timer 2 Interrupt Control Register						
ADR/SIZ	$FFF4201A (8 bits)						
BIT	15	14	13	12	11	10	0	8
FIELD			INT	IEN	ICLR	IL2	IL1	IL0
OPER			R	R/W	C	R/W	R/W	R/W
RESET			0 PL	0 PL	0 PL	0 PL	0 PL	0 PL

This is an 8-bit read/write register that controls the Tick Timer 2 interrupts. It is located at address $FFF4201A.

IL2-IL0 Interrupt Request Level (bits 2-0)l. These three bits select the interrupt level for Tick Timer 2. Level 0 does not generate an interrupt.

ICLR Interrupt Clear. Writing a logic 1 into this bit clears the INT status bit. This bit is always read as zero.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

INT Interrupt Status. When this bit is high a Tick Timer 2 interrupt is being generated at the level programmed in IL2-IL0 (if nonzero). This bit is edge-sensitive and can be cleared by writing a logic 1 into the ICLR control bit.

Tick Timer 1 Interrupt Control Register

REG	Tick Timer 1 Interrupt Control Register							
ADR/SIZ	$FFF4201B (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD			INT	IEN	ICLR	IL2	IL1	IL0
OPER			R	R/W	C	R/W	R/W	R/W
RESET			0 PL	0 PL	0 PL	0 PL	0 PL	0 PL

This is an 8-bit read/write register that controls the Tick Timer 1 interrupts. It is located at address $FFF4201B.

IL2-IL0 Interrupt Request Level (bits 2-0). These three bits select the interrupt level for Tick Timer 1. Level 0 does not generate an interrupt.

ICLR Interrupt Clear. Writing a logic 1 into this bit clears the INT status bit. This bit is always read as zero.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

INT Interrupt Status. When this bit is high a Tick Timer 1 interrupt is being generated at the level programmed in IL2-IL0 (if nonzero). This bit is edge-sensitive and can be cleared by writing a logic 1 into the ICLR control bit.

SCC Error Status Register and Interrupt Control Registers

This section provides addresses and bit level descriptions of the SCC interrupt control registers and status registers.

SCC Error Status Register

REG	SCC Error Status Register							
ADR/SIZ	$FFF4201C (8 bits)							
BIT	31	30	29	28	27	26	25	24
FIELD				RTRY	PRTY	EXT	LTO	SCLR
OPER				R	R	R	R	W/R-0
RESET				0 PL	0 PL	0 PL	0 PL	0

The SCC Error Status Register is located at address $FFF4201C. This 8-bit register (5 bits used) controls the SCC status functions.

SCLR Status Clear. Writing a 1 to this bit clears bits 25 through 28 (LTO, ECT, PRTY, and RTRY). Reading this bit always yields 0.

LTO,BEXT, RTY,RTRY These bits (LTO, BEXT, RTY, and RTRY) indicate the status of the last local peripheral bus error condition encountered by the SCC while performing DMA accesses to the local peripheral bus. A local peripheral bus error condition is flagged by the assertion of TEA*. When the SCC receives TEA* if the source of the error is the local peripheral bus timeout, then LTO is set and EXT, PRTY, and RTRY are cleared. If the source of the TEA* is due to an error in going to the VMEbus, then EXT is set and the other three status bits are cleared. If the source of the error is DRAM parity check error, then PRTY is set and the other three status bits are cleared. If the source of the TEA* is because a retry was needed, then RTRY is set and the other three status bits are cleared. If the source of the error is none of the above conditions, then all four bits are cleared. Writing a 1 to bit 24 (SCLR) also clears all four bits.

SCC Modem Interrupt Control Register

REG	SCC Modem Interrupt Control Register							
ADR/SIZ	$FFF4201D (8 bits)							
BIT	23	22	21	20	19	18	17	16
FIELD			IRQ	IEN	AVEC	IL2	IL1	IL0
OPER			R	R/W	R/W	R/W	R/W	R/W
RESET			X	0 PL	0 PL	0 PL	0 PL	0 PL

This is an 8-bit read/write register that controls the SCC Modem Interrupt functions. It is located at address $FFF4201D.

IL2-IL0 Interrupt Request Level (bits 2-0). These three bits select the interrupt level for SCC modem interrupt. Level 0 does not generate an interrupt.

AVEC When this bit is high, the PCCchip2 supplies the interrupt vector to the interrupt handler during an IACK for SCC modem interrupt. When this bit is low, the PCCchip2 obtains the vector from the SCC and passes it to the BusSwitch.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

IRQ Interrupt Status. This status bit reflects the state of the SCC-IRQ1 pin of the CD2401 (qualified by the IEN bit). When this bit is high, an SCC modem interrupt is being generated at the level programmed in IL2-IL0 (if nonzero). This status bit does not need to be cleared, because it is not edge-sensitive.

SCC Transmit Interrupt Control Register

REG	SCC Transmit Interrupt Control Register							
ADR/SIZ	$FFF4201E (8 bits)							
BIT	15	14	13	12	11	10	0	8
FIELD			IRQ	IEN	AVEC	IL2	IL1	IL0
OPER			R	R/W	R/W	R/W	R/W	R/W
RESET			X	0 PL	0 PL	0 PL	0 PL	0 PL

This is an 8-bit read/write register that controls the SCC Transmit Interrupt functions. It is located at address $FFF4201E.

IL2-IL0 Interrupt Request Level (bits 2-0). These three bits select the interrupt level for SCC Transmit Interrupt. Level 0 does not generate an interrupt.

AVEC When this bit is high, the PCCchip2 supplies the interrupt vector to the BusSwitch during an IACK for SCC transmit interrupt. When this bit is low, the PCCchip2 obtains the vector from the SCC and passes it to the BusSwitch.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

IRQ Interrupt Status. This status bit reflects the state of the SCC-IRQ2 pin of the CD2401 (qualified by the IEN bit). When this bit is high, an SCC Transmit interrupt is being generated at the level programmed in IL2-IL0 (if nonzero). This status bit does not need to be cleared, because it is not edge-sensitive.

SCC Receive Interrupt Control Register

REG	SCC Receive Interrupt Control Register							
ADR/SIZ	$FFF4201F (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	SC1	SC0	IRQ	IEN	AVEC	IL2	IL1	IL0
OPER	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W
RESET	0 PL	0 PL	X	0 PL	0 PL	0 PL	0 PL	0 PL

The SCC Receive Interrupt Control Register is located at address $FFF4201F. This 8-bit register controls the following pin functions.

IL2-IL0 Interrupt Request Level (bits 2-0). These three bits select the interrupt level for SCC Receive Interrupt. Level 0 does not generate an interrupt.

AVEC When this bit is high, the PCCchip2 supplies the interrupt vector to the BusSwitch during an IACK for SCC receive interrupt. When this bit is low, the PCCchip2 obtains the vector from the SCC and passes it to the BusSwitch.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

IRQ Interrupt Status. This status bit reflects the state of the SCC-IRQ3 pin of the CD2401 (qualified by the IEN bit). When this bit is high, an SCC receive interrupt is being generated at the level programmed in IL2-IL0 (if nonzero). This status bit does not does not need to be cleared, because it is not edge-sensitive.

SC1-SC0 Snoop Control (bits 1-0). These control bits determine the value that the PCCchip2 drives onto the local peripheral bus (MC68040 compatible bus) SC1 and SC0 pins, when the CL-CD2401(SCC) performs DMA accesses. During SCC DMA, when bit SC0 is 0, the local peripheral bus pin SC0 is low, and when bit SC0 is 1, the local peripheral bus pin SC0 is high. (The snoop control signals SC1-SC0 are not applicable to any of the MVME197 module series).

Modem PIACK Register

REG	Modem PIACK Register							
ADR/SIZ	$FFF42023 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	MIV7	MIV6	MIV5	MIV4	MIV3	MIV2	MIV1	MIV0
OPER	R	R	R	R	R	R	R	R
RESET	X	X	X	X	X	X	X	X

The Modem PIACK Register is used to execute modem pseudo interrupt acknowledge cycles to the CD2401. When the local peripheral bus master initiates a read cycle to this register, the PCCchip2 executes an interrupt acknowledge cycle to the CD2401 with A7-A0 = $01. (Note that the PILR1 register in the CD2401 should be set to the same value ($01) for the interrupt acknowledge cycle to operate properly). To finish the local peripheral bus read cycle, the PCCchip2 drives the vector received from the CD2401 onto the local peripheral bus data bus, and asserts TA*. Reads to this register are termed pseudo interrupt acknowledge cycles because they are normal read cycles on the local peripheral bus side but they are interrupt acknowledge cycles on the CD2401 side of the PCCchip2. They are necessary to support polled mode operation with the CD2401.

MIV7-MIV0 Modem Interrupt Vector (bits 7-0). These eight bits reflect the modem interrupt vector driven by the CD2401 to the PCCchip2 during a pseudo interrupt acknowledge cycle.

Transmit PIACK Register

REG	Transmit PIACK Register							
ADR/SIZ	$FFF42025 (8 bits)							
BIT	23	22	21	20	19	18	17	16
FIELD	TIV7	TIV6	TIV5	TIV4	TIV3	TIV2	TIV1	TIV0
OPER	R	R	R	R	R	R	R	R
RESET	X	X	X	X	X	X	X	X

The Transmit PIACK Register is used to execute transmit pseudo interrupt acknowledge cycles to the CD2401. When the local peripheral bus master initiates a read cycle to this register, the PCCchip2 executes an interrupt acknowledge cycle to the CD2401 with A7-A0 = $02. (Note that the PILR1 register in the CD2401 should be set to the same value ($02) for the interrupt acknowledge cycle to operate properly). To finish the local peripheral bus read cycle, the PCCchip2 drives the vector received from the CD2401 onto the local peripheral bus data bus, and asserts TA*. Reads to this register are termed pseudo interrupt acknowledge cycles because they are normal read cycles on the local peripheral bus side but they are interrupt acknowledge cycles on the CD2401 side of the PCCchip2. They are necessary to support polled mode operation with the CD2401.

TIV7-TIV0 Transmit Interrupt Vector (bits 7-0). These eight bits reflect the transmit interrupt vector driven by the CD2401 to the PCCchip2 during a pseudo interrupt acknowledge cycle.

Receive PIACK Register

REG	Receive PIACK Register							
ADR/SIZ	$FFF42027 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	RIV7	RIV6	RIV5	RIV4	RIV3	RIV2	RIV1	RIV0
OPER	R	R	R	R	R	R	R	R
RESET	X	X	X	X	X	X	X	X

The Receive PIACK Register is used to execute receive pseudo interrupt acknowledge cycles to the CD2401. When the local peripheral bus master initiates a read cycle to this register, the PCCchip2 executes an interrupt acknowledge cycle to the CD2401 with A7-A0 = $03. (Note that the PILR1 register in the CD2401 should be set to the same value ($03) for the interrupt acknowledge cycle to operate properly). To finish the local peripheral bus read cycle, the PCCchip2 drives the vector received from the CD2401 onto the local peripheral bus data bus, and asserts TA*. Reads to this register are termed pseudo interrupt acknowledge cycles because they are normal read cycles on the local peripheral bus side but they are interrupt acknowledge cycles on the CD2401 side of the PCCchip2. They are necessary to support polled mode operation with the CD2401.

RIV7-RIV0 Receive Interrupt Vector (bits 7-0). These eight bits reflect the transmit interrupt vector driven by the CD2401 to the PCCchip2 during a pseudo interrupt acknowledge cycle.

LANC Error Status and Interrupt Control Registers

This section provides addresses and bit level descriptions of the LANC interrupt control registers and status register.

LANC Error Status Register

REG	LANC Error Status Register							
ADR/SIZ	$FFF42028 (8 bits)							
BIT	31	30	29	28	27	26	25	24
FIELD					PRTY	EXT	LTO	SCLR
OPER					R	R	R	W/R-0
RESET					0 PL	0 PL	0 PL	0

The LANC Error Status Register is located at address $FFF42028. This 8-bit register (4 bits used) controls the LANC error status pin functions.

SCLR Status Clear. Writing a 1 to this bit clears bits 25 through 27 (LTO, ECT, and PRTY). Reading this bit always yields 0.

LTO,EXT, PRTY These bits (LTO, EXT, and PRTY) indicate the status of the last local peripheral bus error condition encountered by the LANC while performing DMA accesses to the local peripheral bus. A local peripheral bus error condition is flagged by the assertion of TEA*. When the LANC receives TEA* if the source of the error is the local peripheral bus timeout, then LTO is set and EXT and PRTY are cleared. If the source of the TEA* is due to an error in going to the VMEbus, then EXT is set and the other two status bits are cleared. If the source of the error is DRAM parity check error, then PRTY is set and the other two status bits are cleared. (The PRTY bit is not applicable to any of the MVME197 module series). If the source of the error is none of the above conditions, then all three bits are cleared. Writing a 1 to bit 24 (SCLR) also clears all three bits.

82596CA LANC Interrupt Control Register

REG	82596CA LANC Interrupt Control Register							
ADR/SIZ	$FFF4202A (8 bits)							
BIT	15	14	13	12	11	10	0	8
FIELD	PLTY	E/L*	INT	IEN	ICLR	IL2	IL1	IL0
OPER	R/W	R/W	R	R/W	C	R/W	R/W	R/W
RESET	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL

The 82596CA LANC Interrupt Control Register is located at address $FFF4202A. This 8-bit register controls the LANC interrupt pin functions.

IL2-IL0 Interrupt Request Level (bits 2-0). These three bits select the interrupt level for the 82596CA LANC. Level 0 does not generate an interrupt.

ICLR Interrupt Clear. In edge-sensitive mode, writing a logic 1 to this bit clears the INT status bit. This bit has no function in level-sensitive mode. This bit is always read as zero.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

INT Interrupt Status. This status bit reflects the state of the INT pin from the LANC (qualified by the IEN bit). When this bit is high, a LANC INT interrupt is being generated at the level programmed in IL2-IL0 (if nonzero).

E/L* Edge/Level Mode. When this bit is high, the interrupt is edge-sensitive. The interrupt is level-sensitive when this bit is low.

PLTY Polarity Mode. When this bit is low, interrupt is activated by a rising edge/high level of the LANC INT pin. When this bit is high, interrupt is activated by a falling edge/low level of the LANC INT pin. Note that if this bit is changed while the E/L* bit is set (or is being set), a LANC interrupt may be generated. This can be avoided by setting the ICLR bit during write cycles that change the E/L* bit.

LANC Bus Error Interrupt Control Register

REG	LANC Bus Error Interrupt Control Register							
ADR/SIZ	$FFF4202B (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	SC1	SC0	INT	IEN	ICLR	IL2	IL1	IL0
OPER	R/W	R/W	R	R/W	C	R/W	R/W	R/W
RESET	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL

This is an 8-bit read/write register that controls the LANC Bus Error Interrupt functions. It is located at address $FFF4202B.

IL2-IL0 Interrupt Request Level (bits 2-0). These three bits select the interrupt level. Level 0 does not generate an interrupt.

ICLR Interrupt Clear. Writing a logic 1 into this bit clears the INT status bit. This bit is always read as zero.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

IRQ Interrupt Status. When this bit is high, a LANC Bus Error interrupt is being generated at the level programmed in IL2-IL0 (if nonzero).

SC1-SC0 Snoop Control (bits 1-0). These two control bits determine the value that the PCCchip2 drives onto the local peripheral bus (MC68040 compatible bus) SC1 and SC0 pins, when the 82596CS(LANC) performs DMA accesses. During LANC DMA, if bit SC0 is 0 then the local peripheral bus pin SC0 is low, and when bit SC0 is 1, the local peripheral pin SC0 is high. (The snoop control signals SC1-SC0 are not applicable to the MVME197 module series).

Programming the SCSI Error Status and Interrupt Registers

This section provides address and bit level description of the SCSI interrupt control register and status register.

SCSI Error Status Register

REG	SCSI Error Status Register							
ADR/SIZ	$FFF4202C (8 bits)							
BIT	31	30	29	28	27	26	25	24
FIELD					PRTY	EXT	LTO	SCLR
OPER					R	R	R	W/R-0
RESET					0 PL	0 PL	0 PL	0

The SCSI Error Status Register is located at address $FFF4202C. This 8-bit register (4 bits used) controls the SCSI error status pins functions.

SCLR Status Clear. Writing a 1 to this bit clears bits 25 through 27 (LTO, ECT, AND PRTY). Reading this bit always yields 0.

LTO,EXT, PRTY These bits (LTO, EXT, and PRTY) indicate the status of the last local peripheral bus error condition encountered by the SCSI processor while performing DMA accesses to the local peripheral bus. A local peripheral bus error condition is flagged by the assertion of TEA*. When the SCSI processor receives TEA* if the source of the error is the local peripheral bus timeout, then LTO is set and EXT and PRTY are cleared. If the source of the TEA* is due to an error in going to the VMEbus, then EXT is set and the other two status bits are cleared. If the source of the error is DRAM parity check error, then PRTY is set and the other two status bits are cleared. (The PRTY bit is not applicable to any of the MVME197 module series). If the source of the error is none of the above conditions, then all three bits are cleared. Writing a 1 to bit 24 (SCLR) also clears all three bits.

SCSI Interrupt Control Register

REG	SCSI Interrupt Control Register							
ADR/SIZ	$FFF4202F (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD			IRQ	IEN		IL2	IL1	IL0
OPER			R	R/W		R/W	R/W	R/W
RESET			0 PL	0 PL		0 PL	0 PL	0 PL

The SCSI Interrupt Control Register is located at address $FFF4202F. This 8-bit register (5 bits used) controls the SCSI interrupt pin functions.

IL2-IL0 Interrupt Request Level (bits 2-0). These three bits select the interrupt level for the SCSI Processor. Level 0 does not generate an interrupt.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

IRQ Interrupt Status. This status bit reflects the state of the IRQ* pin of the SCSI Processor (qualified by the IEN bit). When this bit is high, a SCSI processor interrupt is being generated at the level programmed in IL2-IL0 (if nonzero). This status bit does not need to be cleared, because it is not edge-sensitive.

Programming the Printer Port

This section provides addresses and bit level descriptions of the printer port control, status, and data registers.

Printer ACK Interrupt Control Register

REG	Printer ACK Interrupt Control Register							
ADR/SIZ	$FFF42030 (8 bits)							
BIT	31	30	29	28	27	26	25	24
FIELD	PLTY	E/L*	INT	IEN	ICLR	IL2	IL1	IL0
OPER	R/W	R/W	R	R/W	C	R/W	R/W	R/W
RESET	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL

The Printer ACK Interrupt Control Register is located at address $FFF42030. This 8-bit register controls the printer ACK interrupt pin functions.

IL2-IL0 Interrupt Request Level (bits 2-0). These three bits select the interrupt level for the printer ACK. Level 0 does not generate an interrupt.

ICLR Interrupt Clear. In edge-sensitive mode, writing a logic 1 to this bit clears the INT status bit. This bit has no function in level-sensitive mode. This bit is always read as zero.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

INT Interrupt Status. When this bit is high, a printer ACK interrupt is being generated at the level programmed in IL2-IL0 (if nonzero).

E/L* Edge/Level Mode. When this bit is high, the interrupt is edge-sensitive. The interrupt is level-sensitive when this bit is low.

PLTY Polarity. When this bit is low, interrupt is activated by a falling edge/low level on the PRACKI* pin. When this bit is high, interrupt is activated by a rising edge/high level on the PRACKI* pin. Note that if this bit is changed while the E/L* bit is set (or is being set), an ACK interrupt may be generated. This can be avoided by setting the ICLR bit during write cycles that change the E/L* bit.

Printer FAULT Interrupt Control Register

REG	Printer FAULT Interrupt Control Register							
ADR/SIZ	$FFF42031 (8 bits)							
BIT	23	22	21	20	19	18	17	16
FIELD	PLTY	E/L*	INT	IEN	ICLR	IL2	IL1	IL0
OPER	R/W	R/W	R	R/W	C	R/W	R/W	R/W
RESET	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL

The Printer FAULT Interrupt Control Register is located at address $FFF42031. This 8-bit register controls the printer FAULT interrupt pin functions.

IL2-IL0 Interrupt Request Level (bits 2-0). These three bits select the interrupt level for the printer FAULT. Level 0 does not generate an interrupt.

ICLR Interrupt Clear. In edge-sensitive mode, writing a logic 1 to this bit clears the INT status bit. This bit has no function in level-sensitive mode. This bit is always read as zero.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

INT Interrupt Status. When this bit is high, a printer FAULT interrupt is being generated at the level programmed in IL2-IL0 (if nonzero).

E/L* Edge/Level Mode. When this bit is high, the interrupt is edge-sensitive. The interrupt is level-sensitive when this bit is low.

PLTY Polarity Mode. When this bit is low, interrupt is activated by a falling edge/low level of the PRFAULTI* pin. When this bit is high, interrupt is activated by a rising edge/high level of the PRFAULTI* pin. Note that if this bit is changed while the E/L* bit is set (or is being set), a FAULT interrupt may be generated. This can be avoided by setting the ICLR bit during write cycles that change the E/L* bit.

Printer SEL Interrupt Control Register

REG	Printer SEL Interrupt Control Register							
ADR/SIZ	$FFF42032 (8 bits)							
BIT	15	14	13	12	11	10	9	8
FIELD	PLTY	E/L*	INT	IEN	ICLR	IL2	IL1	IL0
OPER	R/W	R/W	R	R/W	C	R/W	R/W	R/W
RESET	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL

The Printer SEL Interrupt Control Register is located at address $FFF42032. This 8-bit register controls the printer SEL interrupt pin functions.

IL2-IL0 Interrupt Request Level (bits 2-0). These three bits select the interrupt level for the printer SEL. Level 0 does not generate an interrupt.

ICLR Interrupt Clear. In edge-sensitive mode, writing a logic 1 to this bit clears the INT status bit. This bit has no function in level-sensitive mode. This bit is always read as zero.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

INT Interrupt Status. When this bit is high, a printer SEL interrupt is being generated at the level programmed in IL2-IL0 (if nonzero).

E/L* Edge/Level Mode. When this bit is high, the interrupt is edge-sensitive. The interrupt is level-sensitive when this bit is low.

PLTY Polarity Mode. When this bit is low, interrupt is activated by a rising edge/high level of the SEL pin. When this bit is high, interrupt is activated by a falling edge/low level of the SEL pin. Note that if this bit is changed while the E/L* bit is set (or is being set), a SEL interrupt may be generated. This can be avoided by setting the ICLR bit during write cycles that change the E/L* bit.

Printer PE Interrupt Control Register

REG	Printer PE Interrupt Control Register							
ADR/SIZ	$FFF42033 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	PLTY	E/L*	INT	IEN	ICLR	IL2	IL1	IL0
OPER	R/W	R/W	R	R/W	C	R/W	R/W	R/W
RESET	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL

The Printer PE Interrupt Control Register is located at address $FFF42033. This 8-bit register controls the printer PE interrupt pin functions.

IL2-IL0 Interrupt Request Level (bits 2-0). These three bits select the interrupt level for the printer PE. Level 0 does not generate an interrupt.

ICLR Interrupt Clear. In edge-sensitive mode, writing a logic 1 to this bit clears the INT status bit. This bit has no function in level-sensitive mode. This bit is always read as zero.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

INT Interrupt Status. When this bit is high, a printer PE interrupt is being generated at the level programmed in IL2-IL0 (if nonzero).

E/L* Edge/Level Mode. When this bit is high, the interrupt is edge-sensitive. The interrupt is level-sensitive when this bit is low.

PLTY Polarity Mode. When this bit is low, interrupt is activated by a rising edge/high level of the PE pin. When this bit is high, interrupt is activated by a falling edge/low level of the PE pin. Note that if this bit is changed while the E/L* bit is set (or is being set), a PE interrupt may be generated. This can be avoided by setting the ICLR bit during write cycles that change the E/L* bit.

Printer BUSY Interrupt Control Register

REG	Printer BUSY Interrupt Control Register							
ADR/SIZ	$FFF42034 (8 bits)							
BIT	31	30	29	28	27	26	25	24
FIELD	PLTY	E/L*	INT	IEN	ICLR	IL2	IL1	IL0
OPER	R/W	R/W	R	R/W	C	R/W	R/W	R/W
RESET	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL	0 PL

The Printer BUSY Interrupt Control Register is located at address $FFF42034. This 8-bit register controls the printer BUSY interrupt pin functions.

IL2-IL0 Interrupt Request Level (bits 2-0). These three bits select the interrupt level for the printer BUSY. Level 0 does not generate an interrupt.

ICLR Interrupt Clear. In edge-sensitive mode, writing a logic 1 to this bit clears the INT status bit. This bit has no function in level-sensitive mode. This bit is always read as zero.

IEN Interrupt Enable. When this bit is high, the interrupt is enabled. The interrupt is disabled when this bit is low.

INT Interrupt Status. When this bit is high, a printer BUSY interrupt is being generated at the level programmed in IL2-IL0 (if nonzero).

E/L* Edge/Level Mode. When this bit is high, the interrupt is edge-sensitive. The interrupt is level-sensitive when this bit is low.

PLTY Polarity Mode. When this bit is low, interrupt is activated by a rising edge/high level of the BUSY pin. When this bit is high, interrupt is activated by a falling edge/low level of the BUSY pin. Note that if this bit is changed while the E/L* bit is set (or is being set), a BUSY interrupt may be generated. This can be avoided by setting the ICLR bit during write cycles that change the E/L* bit.

Printer Input Status Register

REG	Printer Input Status Register							
ADR/SIZ	$FFF42036 (8 bits)							
BIT	15	14	13	12	11	10	0	8
FIELD	PINT			ACK	FLT	SEL	PE	BSY
OPER	R			R	R	R	R	R
RESET	X			X	X	X	X	X

The Printer Input Status Register is located at address $FFF42036. This 8-bit register controls the printer input status pin functions.

BSY Printer Busy. This bit reflects the state of the Printer Busy input pin. It is 1 when BSY is high and 0 when BSY is low.

PE Printer Paper Error. This bit reflects the state of the Printer Paper Error input pin. It is 1 when PE is high and 0 when PE is low.

SEL Printer Select. This bit reflects the state of the Printer Select input pin. It is 1 when SELECT is high and 0 when SELECT is low.

FLT Printer Fault. This bit reflects the state of the Printer Fault input pin. It is 1 when FAULT* is low and 0 when FAULT* is high.

ACK Printer Acknowledge. This bit reflects the state of the Printer Acknowledge input pin. It is 1 when ACK* is low and 0 when ACK* is high.

PINT Printer Interrupt Status. When this bit is high, an interrupt is being generated at the level programmed in the Printer Interrupt Level Register. The interrupt may come from one or more printer status pins.

Printer Port Control Register

REG	Printer Port Control Register							
ADR/SIZ	$FFF42037 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD				DOEN	INP	STB	FAST	MAN
OPER				R/W	R/W	R/W	R/W	R/W
RESET				0 PL	0 PL	0 PL	0 PL	0 PL

The Printer Port Control Register is located at address $FFF42037. This 8-bit register (5 bits used) controls the printer port pin functions.

MAN Manual Strobe Control. This bit selects the auto or manual mode for the printer strobe. When this bit is low, the printer strobe is generated automatically by a write to the printer data register (auto mode). When this bit is high, the strobe pin is directly controlled by the STB control bit (manual mode).

FAST Strobe Timing. In auto mode, this bit controls the printer strobe timing. When this bit is low, the strobe time is 212 BCLK periods (10.6 µsec at 20MHz, 8.5 µsec at 25MHz, and 6.4 µsec at 33MHz). When this bit is high, the strobe time is 50 BCLK periods (2.5 µsec at 20MHz, 2 µsec at 25MHz, and 1.5 µsec at 33MHz). Note that the strobe time is the width of the low-going pulse generated on the STB* pin. Also note, that after a write to the printer data register, the PCCchip2 delays about one strobe time before issuing the STB* pulse. This bit is not used in manual mode.

STB Manual Strobe Control. In the manual mode, the software controls the strobe timing. When this bit is high, the printer strobe is activated. When this bit is low, the printer strobe is not activated. This bit has no function in auto mode.

INP Printer Input Prime. This bit controls the input prime signal. When this bit is high, the input prime signal is activated. When this bit is low, the input prime signal is not activated. Software must control the timing of the printer input prime signal.

DOEN Printer Data Output Enable. This bit controls the external data buffer for the printer port. When this bit is high, the external printer data buffer is enabled. When this bit is low, the external printer data buffer is disabled. For normal connection to a printer, DOEN should be set to 1.

Chip Speed Register

REG	Chip Speed Register															
ADR/SIZ	$FFF42038 (16 bits)															
BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	CS31-CS16															
OPER	R															
RESET	X															

This is a 16-bit read register that controls the Chip Speeds. It is located at address $FFF42038.

CS31-CS16 Chip Speed (bits 31-16). This status bits help determine the relative speed of the silicon used to implement each individual chip. For a given BCLK frequency and duty cycle, the value and chip speeds shown in the chart below are valid. Notice that BCLK is half of the frequency of the MC88110 input clock.

Value of CS31-CS16	Chip Speed
$0000	Slowest
$0001	Next Slowest
$0003	2nd Next Slowest
$0007	3rd Next Slowest
$1FFF	3rd Next Fastest
$3FFF	2nd Next Fastest
$7FFF	Next Fastest
$FFFF	Fastest

Note

The register must be read as a double byte at address $FFF42038 in order to have meaningful information. If it is accessed as 4 bytes, then all of its bits yield 0's.

Printer Data Register

REG	Printer Data Register															
ADR/SIZ	$FFF4203A (16 bits)															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PD15-PD0															
OPER	R/W															
RESET	X															

This is a 16-bit read/write register that controls the Printer Data functions. It is located at address $FFF4203A.

PD15-PD0 Printer Data (bits 15-0). Writing to these bits causes the PCCchip2 to latch data into the external printer data buffer. Generally the printer data buffer only connects to PD7-PD0, because most printer data paths are 8 bits wide. PD7-PD0 can be accessed as an 8-bit register at location $FFF4203B, or PD15-PD0 can be accessed as a 16-bit register at location $FFF4203A. In auto mode, writing these bits also generates the strobe for the printer. Reading these bits causes the PCCchip2 to read the data from the printer data signal lines (no strobe is generated). When the DOEN bit is set, the printer data signal lines are driven by the external printer data buffer. When the DOEN bit is cleared, they must be terminated to high or to low and/or an external device must drive them.

Interrupt Priority Level Register

REG	Interrupt Priority Level Register							
ADR/SIZ	$FFF4203E (8 bits)							
BIT	15	14	13	12	11	10	9	8
FIELD						IPL2	IPL1	IPL0
OPER						R	R	R
RESET						X	X	X

(This register is not applicable to any of the MVME197 module series).

The Interrupt Priority Level Register is located at address $FFF4203E. This 8-bit register (3 bits used) defines the interrupt priority level pin functions.

IPL2-IPL0 Interrupt Priority Level (bits 2-0). These three bits reflect the priority-encoded interrupt request level. This level is a combination of the PCCchip2 interrupt requests and the interrupt requests driven onto the EIPL2-EIPL0 pins. Note that when the C040 bit is cleared, external devices can drive EIPL2-EIPL0 with their interrupt requests. When C040 is set, the PCCchip2 drives EIPL2-EIPL0 with its interrupt requests. In this case (C040 set), IPL2-IPL0 only reflect PCCchip2 interrupt requests. The IPL bits are encoded as shown below:

IPL2	IPL1	IPL0	Priority Level	Comments
0	0	0	0	No Interrupt
0	0	1	1	Lowest Level
0	1	0	2	▲
0	1	1	3	
1	0	0	4	
1	0	1	5	
1	1	0	6	▼
1	1	1	7	Highest Level

Interrupt Mask Level Register

REG	Interrupt Mask Level Register							
ADR/SIZ	$FFF4203F (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD						MSK2	MSK1	MSK0
OPER						R/W	R/W	R/W
RESET						1 PL	1 PL	1 PL

(This register is not applicable to any of the MVME197 module series).

The Interrupt Mask Level Register is located at address $FFF4203F. This 8-bit register (3 bits used) defines the interrupt mask level pin functions.

MSK2-MSK0 Interrupt Mask Level (bits 2-0). The interrupt mask level bits determine the level which must be exceeded by IPL2-IPL0 in order for the PCCchip2 to assert its INT pin. The MSK bits are encoded as follows:

MSK2	MSK1	MSK0	Priority Level	Comments
0	0	0	0	Lowest Level
0	0	1	1	▲
0	1	0	2	
0	1	1	3	
1	0	0	4	
1	0	1	5	
1	1	0	6	▼
1	1	1	7	Highest Level

BUSSWITCH 4

Introduction

This document describes the architecture and usage of the MC88110 BusSwitch. The BusSwitch essentially provides a bridge between the MC88110 bus (Processor Bus) and a MC68040 compatible bus (Local Peripheral Bus). This allows the MC88110 access to MC68040 peripherals in a very efficient way. It also simplifies interfacing to the MC88110 in general, since it is easier to interface to a 32-bit data bus than to the MC88110's 64-bit data bus.

BusSwitch Features

These are some of the features of the MC88110 BusSwitch:

- ❑ Provides a 64-bit master/slave port on the processor bus and a 32-bit master/slave port on the local peripheral bus
- ❑ Provides bus arbitration logic for the processor bus
- ❑ Provides ROM/Flash Memory decode logic and control for a 4 megabyte space
- ❑ Seven level interrupt handler circuit to prioritize interrupts
- ❑ Supports up to two MC88110 microprocessors on the processor bus (this includes the MC88110 MPU and MC88410 Cache Controller combination for the MVME197DP and MVME197SP modules of the MVME197 series of Single Board Computers)
- ❑ Programmable processor bus and local peripheral bus map decoders
- ❑ Write post buffers in the processor bus and local peripheral bus
- ❑ Two 32-bit tick timers
- ❑ Provides four 32-bit general purpose registers for software use
- ❑ Cross processor interrupt capability for sending messages between processors
- ❑ Provides decoding for external cache flush control logic for the MC88410 cache controller
- ❑ JTAG boundary scan interface

System Overview

The BusSwitch provides an interconnect path between the MC88110 bus (referred to as the *Processor Bus*), and an MC68040 compatible bus (referred to as the *Local Peripheral Bus*). This is desirable from a systems design standpoint, since 32-bit peripherals used in existing MC68040 systems can be used with the MC88110. This includes custom gate arrays such as the VMEchip2, which can be used with the BusSwitch to allow the MC88110 access to the VMEbus. Figure 4-1 shows a simplified block diagram of an MC88110 system with the BusSwitch interface. Figure 4-2 provides a simplified block diagram for the BusSwitch.

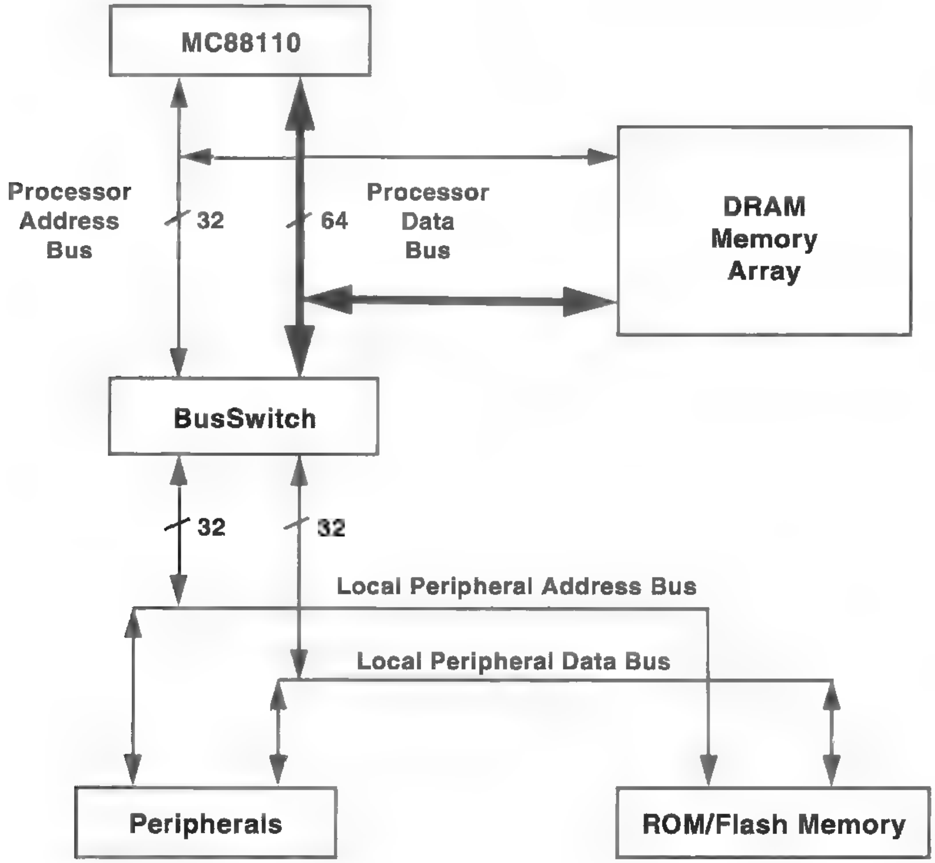


Figure 4-1. Simplified MC88110 System Block Diagram

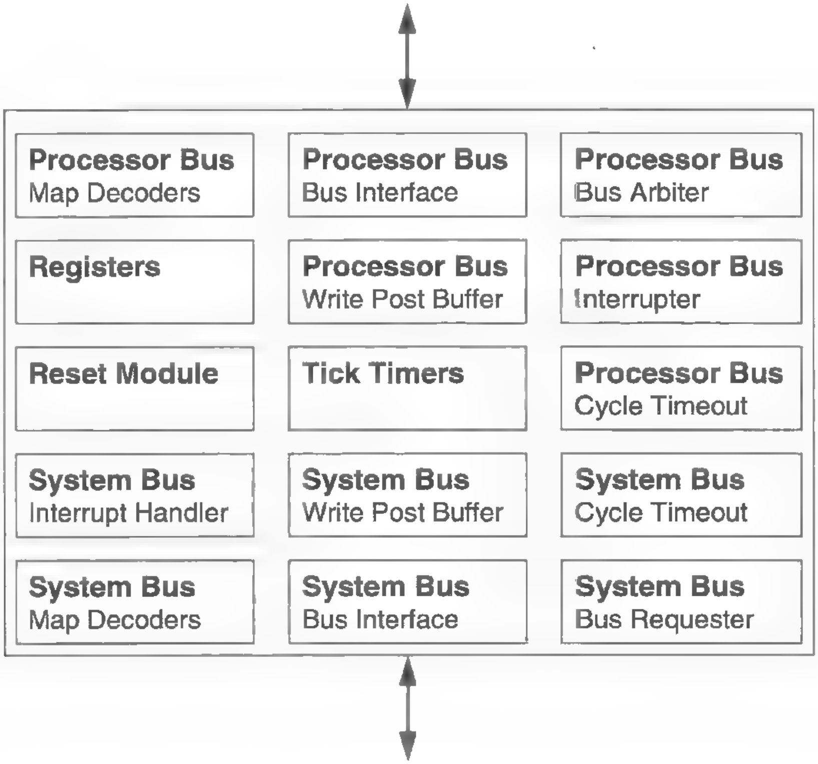


Figure 4-2. BusSwitch Block Diagram

Processor Bus Functions

Processor Bus Arbiter

The processor bus arbiter selects the next master on the processor *address* bus among all the contending requesters. Potential requesters of this bus include the MC88110, the BusSwitch itself on behalf of another master on the local peripheral bus, a DMA controller, or even a second MC88110. The processor bus arbiter supports three external bus masters. The first MC88110 is connected to BR0* and is called CPU0. If another MC88110 is used, it is connected to BR1* and is called CPU1. BR2* is reserved for a non-CPU type master, like a peripheral DMA controller. The priorities for BR2*, BR1*, and BR0* can be changed, but at power on/reset they are set from highest to lowest as BR2*, BR1*, and BR0*. Two bus arbitration algorithms are supported:

Fixed Priority This is the default algorithm. The bus priority assignments are fixed. After RESET, they are set from high to low as follows: BR2*, BR1*, and BR0*. The APRI field in the GCSR register can be used to change the highest priority level.

Round Robin The bus priority assignments rotate after each arbitration cycle, from BR(n)* to BR(n-1)*.

In addition, the processor bus arbiter supports bus parking, in which the bus grant remains asserted after the bus request is removed. This reduces the number of arbitrations required in cases where one master generates the majority of the bus transactions. At power on/reset BR1* and BR2* are disabled. This ensures that only the processor at BG0* gets access to the bus after power on/reset. The processor at BG0* (CPU0) should enable the BR1* and BR2* signals when it is safe to do so by setting the BREN bit in the GCSR. The DBG* (Data Bus Grant) signal of each MC88110 must be grounded, since there is no data bus arbiter.

Processor Bus Decoder

The processor bus decoding is implemented with multiple decoders which include:

- BusSwitch Registers decoder
- Programmable processor to local peripheral bus decoders
- ROM/Flash memory decoder
- External ROM decoder
- Invalidate Decoder

The BusSwitch register decoder decodes all the internal registers of the BusSwitch, which includes various byte, word, and longword sized registers. This address space starts at $FFF00000. Registers can be accessed with byte, word, longword, or double accesses. Line transfers are not allowed. For more information refer to the BusSwitch registers section.

The BusSwitch includes four programmable processor bus map decoders, which control access to the local peripheral bus from the processor bus. These decoders provide a window into the local peripheral bus from the processor bus. The most significant 16 bits of the processor address bus are compared with the address range of each map decoder, and if the address falls within the range of one of the map decoders, a bus request signal to the local peripheral bus is issued. For each map, there is an associated set of attributes. Each map decoder contains the following registers:

Processor Start Address Register

Processor End Address Register

Processor Translation Register

Processor Translation Select Register

Processor Attributes Register

The ROM decoder and related control logic, decodes a 4 megabyte address space starting at $FF800000 used for Flash Memory. The ROMCR register is used to control this decoder.

The external ROM decoder is enabled after reset. In this case any processor access in the 4 megabyte space starting at $00000000 will be decoded and translated to a MC68040 bus cycle. This provides a mechanism for fetching the reset vector. Clearing the ROM0 bit in the ROMCR register disables this decoder.

The Invalidate Decoder decodes Invalidate Cycles regardless of the address, and terminates them by asserting PTA* and TA*. Invalidate cycles are generated by the MC88110, when shared cache entries are modified to maintain cache coherency with external caches. The INVD bit in the GCSR register must be set to enable this decoder.

Processor Write Post Buffer

The processor write post buffer stores one byte, word, longword, double longword, or one cache line (4 double longwords) during a processor bus master write cycle along with the address. The cycle is immediately acknowledged, and the processor bus master is free to do something else. In the meantime, the BusSwitch requests use of the local peripheral bus and then completes the transfer. The write post buffer is enabled by the WPEN bit in the PAR (Processor Attribute Register). Note that there are four PAR registers, one for each decoder.

Processor Bus Timer

The Processor Bus Timer (PBT) allows the processor to recover from a lockup condition when a processor resource does not respond to the cycle. The PBT starts ticking when DBB* is asserted, and if the cycle is not terminated (TA*, TEA* or TRTRY* is asserted) before the timeout period, TEA* is asserted by the PBT. This timer is disabled if the access goes to the local peripheral bus. The processor bus timeout is programmable, and the timer can be disabled by software, or by the PTOD* pin.

Processor Interrupter

The Processor Interrupter controls the INT* and NMI* signals to each processor on the processor bus. Most interrupts use the INT* line, but two special cases use the NMI* line.

INT* Interrupts

Interrupts signaled via the INT* line have a 3-bit priority level associated with them, with 7 being the highest and 1 being the lowest. During interrupts, the interrupt level can be read from the LEVEL register. Each level has a corresponding vector register associated with it: level*i* → VECTOR*i*. The corresponding vector register is read to obtain the vector. Interrupts can be generated by the IPL lines, the XIPL lines, various internal sources in the BusSwitch, and two external input signals, PALINT* and IRQ*. Interrupts at a specified level can be steered to CPU0 or CPU1 via the ISEL registers (applicable to the MVME197DP only).

NMI* Interrupts

Interrupts signaled via the NMI* line do not have a level associated with them. The processor has to check the INT bits in the ABORT and CPINT registers to find the cause of the interrupt. CPU0 receives ABORT and CPINT interrupts. CPU1 only receives CPINT interrupts.

ABORT	Abort signal.
CPIRQ	Cross Processor Interrupt.

Reset Module

The BusSwitch includes a reset module that is independent of the rest of the BusSwitch. It has three inputs (RSTPON*, RSTXI*, and RSTSWI*) that are debounced and combined to generate a reset outupt (RSTXO*). These inputs are connected to the power up reset signal, the VMEchip2 reset output, and the reset switch, respectively. The reset output is used as the local board reset signal.

Local Peripheral Bus Functions

System Bus Requester

The system bus requester requests access to the local peripheral bus when required by the current MC88110 transfer. This can be triggered by one of the following decoders:

Processor bus decoder

ROM decoder

External ROM decoder

Interrupt handler

System Bus Timer

The System Bus Timer (SBT) allows the current local peripheral bus master to recover from a lockup condition when a resource on the local peripheral bus does not respond to the cycle. The SBT starts ticking when STS* is asserted, and if STA* is not asserted before the timeout period, STEA* is asserted by the SBT if the access is not write posted. If the access is write posted, a write post error interrupt is generated. The local peripheral bus timeout is programmable, and the timer can be disabled by software, or by the STOD* pin.

System Bus Map Decoder

The BusSwitch includes four programmable system bus map decoders, which control access to the processor bus from the local peripheral bus. The map decoders provide a window into the processor bus from the local peripheral bus. The most significant 16 bits of the local peripheral address bus are compared with the address range of each map decoder, and if the address falls within the range of one of the map decoders, a bus request signal to the processor bus is issued. For each map, there is an associated set of attributes. Each map decoder contains the following registers:

System Start Address Register

System End Address Register

System Address Translation Register

System Attribute Register

Interrupt Handler

The interrupt handler function provides the capability to generate MC68040 IACK cycles, and along with the LEVEL, MASK, and VECTOR[7:1] registers, allows the MC88110 to emulate the vectored interrupt capabilities of the MC68040.

MC88410 Flush/Invalidate Control

(This section is applicable only to the MVME197DP/SP versions of the MVME197 series).

The MC88410 cache controller requires external circuitry for cache flushing and invalidation. The MVME197DP/SP implements this logic with the

BusSwitch and some external logic. The BusSwitch provides the decode signals and bus cycle timing control, and the external logic implements the actual registers necessary for operation. Note that from the user's perspective, these registers reside in the BusSwitch. To enable access to these registers, the XCC bit in the BusSwitch GCSR register must be set. Two registers are required for operation with the MC88410 cache controller:

XCCR	External Cache Control Register
XCFR	External Cache Flush Register

Each MC88110 has its corresponding XCCR and XCFR registers, and can initiate a cache flush/invalidate operation only to its corresponding MC88410. At power up/reset, both MC88410 cache controllers are invalidated.

BusSwitch Registers

The BusSwitch register map is shown in Table 4-1.

Conventions

The following conventions are used in the following BusSwitch register charts.

R	Read only field.
R/W	Read/Write field.
S	Writing a one to this field sets this field or another field.
C	Writing a one to this field clears this field or another field.
P	This field is initialized only at power-on reset.

Table 4-1. BusSwitch Register Memory Map

BusSwitch Base Address = $FFF00000

<table>
<tr><th>Offset</th><th>63 56</th><th>55 48</th><th>47</th><th>32</th><th>31</th><th>16</th><th>15</th><th>0</th></tr>
<tr><td>0</td><td>CHIPID</td><td>CHIPREV</td><td colspan="2">GCSR</td><td colspan="2">IODATA</td><td colspan="2">IODIR</td></tr>
<tr><td>8</td><td colspan="2">PSAR1</td><td colspan="2">PEAR1</td><td colspan="2">PSAR2</td><td colspan="2">PEAR2</td></tr>
<tr><td>10</td><td colspan="2">PSAR3</td><td colspan="2">PEAR3</td><td colspan="2">PSAR4</td><td colspan="2">PEAR4</td></tr>
<tr><td>18</td><td colspan="2">PTR1</td><td colspan="2">PTSR1</td><td colspan="2">PTR2</td><td colspan="2">PTSR2</td></tr>
<tr><td>20</td><td colspan="2">PTR3</td><td colspan="2">PTSR3</td><td colspan="2">PTR4</td><td colspan="2">PTSR4</td></tr>
<tr><td>28</td><td colspan="2">SSAR1</td><td colspan="2">SEAR1</td><td colspan="2">SSAR2</td><td colspan="2">SEAR2</td></tr>
<tr><td>30</td><td colspan="2">SSAR3</td><td colspan="2">SEAR3</td><td colspan="2">SSAR4</td><td colspan="2">SEAR4</td></tr>
<tr><td>38</td><td colspan="2">STR1</td><td colspan="2">STSR1</td><td colspan="2">STR2</td><td colspan="2">STSR2</td></tr>
<tr><td>40</td><td colspan="2">STR3</td><td colspan="2">STSR3</td><td colspan="2">STR4</td><td colspan="2">STSR4</td></tr>
<tr><td>48</td><td>PAR1</td><td>PAR2</td><td>PAR3</td><td>PAR4</td><td>SAR1</td><td>SAR2</td><td>SAR3</td><td>SAR4</td></tr>
<tr><td>50</td><td></td><td>BTIMER</td><td>PADJUST</td><td>PCOUNT</td><td colspan="4">PAL</td></tr>
<tr><td>58</td><td colspan="4">WPPA</td><td colspan="2">WPTPA</td><td>WPPAT</td><td></td></tr>
<tr><td>60</td><td colspan="2">ROMCR</td><td>TCTRL1</td><td>TCTRL2</td><td>LEVEL</td><td>MASK</td><td>ISEL0</td><td>ISEL1</td></tr>
<tr><td>68</td><td>ABORT</td><td>CPINT</td><td>TINT1</td><td>TINT2</td><td>WPINT</td><td>PALINT</td><td>XINT</td><td>VBASE</td></tr>
<tr><td>70</td><td colspan="4">TCOMP1</td><td colspan="4">TCOUNT1</td></tr>
<tr><td>78</td><td colspan="4">TCOMP2</td><td colspan="4">TCOUNT2</td></tr>
<tr><td>80</td><td colspan="4">GPR1</td><td colspan="4">GPR2</td></tr>
<tr><td>88</td><td colspan="4">GPR3</td><td colspan="4">GPR4</td></tr>
<tr><td>90</td><td colspan="4">XCTAGS</td><td colspan="4"></td></tr>
</table>

<table>
<tr><td>100</td><td>XCCR</td><td>VECTOR1</td></tr>
<tr><td>108</td><td>VECTOR2</td><td>VECTOR3</td></tr>
<tr><td>110</td><td>VECTOR4</td><td>VECTOR5</td></tr>
<tr><td>118</td><td>VECTOR6</td><td>VECTOR7</td></tr>
</table>

Base Address = $FF8XXXXX (MVME197DP and MVME197SP only)

XCFR

Chip ID Register - CHIPID

REG	Chip ID Register		
ADR/SIZ	$FFF00000 (8 bits)		
BIT	7	--	0
FIELD	CHIPID		
OPER	R		
RESET	$21		

The CHIPID is an 8-bit read-only register with a unique ID code. All writes to this register are ignored, but they will be terminated with TA*.

Chip Revision Register - CHIPREV

REG	Chip Revision Register		
ADR/SIZ	$FFF00001 (8 bits)		
BIT	7	--	0
FIELD	CHIPREV		
OPER	R		
RESET	*		

The CHIPREV is an 8-bit read-only register hardwired to reflect the revision level of the BusSwitch ASIC. All writes to this register are ignored, but they will be terminated with TA*.

General Control and Status Register - GCSR

REG	General Control and Status Register							
ADR/SIZ	$FFF00002 (8 bits of 16) (High Byte)							
BIT	15	14	13	12	11	10	0	8
FIELD	POR	TBB	TDPR		INCB	XCC	TCPU1	XIPL
OPER	R	R/W	R/W	R	R/W	R/W	R/W	R/W
RESET	*	0	0	0	0	0	0	0

REG	General Control and Status Register							
ADR/SIZ	$FFF00003 (8 bits of 16)(Low Byte)							
BIT	7	6	5	4	3	2	1	0
FIELD	USR	INVD	B410	CPUID	BREN	AMOD	APRI	
OPER	R/W	R/W	R	R	R/W	R/W	R/W	
RESET	0	0	*	0	0	0	3	

This register provides control and status information of various BusSwitch functions.

APRI Arbitration Priority. This field specifies which bus request level has the highest priority. It only has meaning when the AMODE bit is 0.

AMOD Arbitration Mode. If cleared, fixed priority arbitration is used, and the APRI field specifies the highest priority requester. If set, round robin arbitration is used, and the highest priority rotates after each arbitration cycle, from BR(n) to BR(n-1).

BREN Bus Request Enable. If set, the bus request signals BR0*, BR1*, and BR2* are received by the internal bus arbiter. If cleared, only BR0* can be received. Since this bit is cleared after reset, only CPU0 can get access to the bus.

CPUID CPU ID. This bit is cleared if the CPU at BR0* is the processor bus master. This bit is set if the CPU at BR1* is the current bus master.

B410 BUS410. This bit indicates the BusSwitch processor bus operating mode. If cleared, the bus is MC88110 compatible. If set, the bus is MC88410 compatible. PTA* is used to select the bus mode. If PTA* is low at reset, B410 will be set.

INVD Invalidate Decoder. This bit enables the Invalidate Cycle decoder if set. When enabled, all invalidate cycles are terminated with TA*.

USR User Enable. If cleared, only supervisor data accesses to the BusSwitch registers is allowed. If set, user data accesses are also allowed.

XIPL External IPL Enable. If cleared, I/O port bits IOP[8:6] operate as regular I/O port bits. If set, IOP[8:6] are used as additional IPL signals, assigned as XIPL[2:0]. (The MVME197 module series connect the IPL outputs from the VMEchip2 to these XIPL lines. The XIPL bit should be set).

TCPU1 Test CPU 1 Registers. This bit is used only if TDPR is set. If TCPU1 is cleared, the CPU0 DP registers can be accessed. If set, the CPU1 DP registers can be accessed.

XCC External Cache Controller. If cleared, I/O port bits IOP[2:0] operates as regular I/O port bits. If set, IOP[2:0] are used to enable flush and invalidation control for an external MC88410 cache controller. As a side effect, flash memory writes are disabled while this bit is set. (This feature is applicable only to the MVME197DP/SP versions of the MVME197 series).

INCB Increment On Burst. If set, addresses are incremented during burst transfers on the local peripheral bus.

TDPR Test Dual Processor Registers. This bit is used along with the TCPU1 bit to access the DP registers. For normal operation, this bit should be cleared. If set, the DP registers specified by TCPU1 will be selected.

TBB Test Bus Busy. This bit is used for testing the BusSwitch during manufacture, and must be cleared during normal operation.

POR Power On Reset. If set, the last reset was a power on reset.

I/O Data Register - IODATA

<table>
<tr><td><b>REG</b></td><td colspan="8">I/O Data Register</td></tr>
<tr><td><b>ADR/SIZ</b></td><td colspan="8">$FFF00004 (8 bits of 16) (High Byte)</td></tr>
<tr><td><b>BIT</b></td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>0</td><td>8</td></tr>
<tr><td><b>FIELD</b></td><td colspan="7"></td><td>IOP8</td></tr>
<tr><td><b>OPER</b></td><td colspan="7">R</td><td>R/W</td></tr>
<tr><td><b>RESET</b></td><td colspan="7">0</td><td>0</td></tr>
</table>

<table>
<tr><td><b>REG</b></td><td colspan="8">I/O Data Register</td></tr>
<tr><td><b>ADR/SIZ</b></td><td colspan="8">$FFF00005 (8 bits of 16) (Low Byte)</td></tr>
<tr><td><b>BIT</b></td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td><b>FIELD</b></td><td>IOP7</td><td>IOP6</td><td>IOP5</td><td>IOP4</td><td>IOP3</td><td>IOP2</td><td>IOP1</td><td>IOP0</td></tr>
<tr><td><b>OPER</b></td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td><td>R/W</td></tr>
<tr><td><b>RESET</b></td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

The IODATA register is used to read or write to the I/O pins, as specified by the IODIR register.

I/O Direction Register - IODIR

REG	I/O Direction Register							
ADR/SIZ	$FFF00006 (8 bits of 16) (High Byte)							
BIT	15	14	13	12	11	10	0	8
FIELD								IOP8
OPER	R							R/W
RESET	0							0

REG	I/O Direction Register							
ADR/SIZ	$FFF00007 (8 bits of 16) (Low Byte)							
BIT	7	6	5	4	3	2	1	0
FIELD	IOP7	IOP6	IOP5	IOP4	IOP3	IOP2	IOP1	IOP0
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0

The IODIR register is used to control the direction of each port bit in the IODATA register. If a bit is set, the corresponding port is programmed for output. If a bit is cleared, the corresponding port is programmed for input.

Processor Start Address Register - PSAR(1-4)

REG	Processor Start Address Register (1-4)		
ADR/SIZ	PSAR1 - $FFF00008 (16 bits)		
ADR/SIZ	PSAR2 - $FFF0000C (16 bits)		
ADR/SIZ	PSAR3 - $FFF00010 (16 bits)		
ADR/SIZ	PSAR4 - $FFF00014 (16 bits)		
BIT	15	--------------------------------------	0
FIELD	PSAR(1-4)		
OPER	R/W		
RESET	0 P		

The PSAR(1-4) is a 16-bit read/write register used to indicate the starting address of a particular memory area on the local peripheral bus to be accessed from the processor bus. The value loaded in this register is compared with the upper 16 bits of the incoming processor bus address. A local peripheral bus request is generated if the address falls in the range specified by the PSAR(1-4) and its corresponding PEAR(1-4) register. In case of overlapping ranges, a priority scheme selects which decoder will be used, with PSAR1/PEAR1 having the highest priority and PSAR4/PEAR4 having the lowest.

Processor End Address Register - PEAR(1-4)

REG	Processor End Address Register (1-4)		
ADR/SIZ	PEAR1 - $FFF0000A (16 bits)		
ADR/SIZ	PEAR2 - $FFF0000E (16 bits)		
ADR/SIZ	PEAR3 - $FFF00012 (16 bits)		
ADR/SIZ	PEAR4 - $FFF00016 (16 bits)		
BIT	15	- -	0
FIELD	PEAR(1-4)		
OPER	R/W		
RESET	0 P		

The PEAR1 is a 16-bit register used to indicate the ending address of a particular memory area on the local peripheral bus to be accessed from the processor bus. The value loaded in this register is compared with the upper 16 bits of the incoming processor bus address. A local peripheral bus request is generated if the address falls in the range specified by the PSAR1(1-4) and its corresponding PEAR1(1-4) register. In case of overlapping ranges, a priority scheme selects which decoder will be used, with PSAR1/PEAR1 having the highest priority and PSAR4/PEAR4 having the lowest.

Processor Translation Register - PTR(1-4)

REG	Processor Translation Register (1-4)		
ADR/SIZ	PTR1 - $FFF00018 (16 bits)		
ADR/SIZ	PTR2 - $FFF0001C (16 bits)		
ADR/SIZ	PTR3 - $FFF00020 (16 bits)		
ADR/SIZ	PTR4 - $FFF00024 (16 bits)		
BIT	15	- -	0
FIELD	PTR(1-4)		
OPER	R/W		
RESET	0 P		

The PTR(1-4) is a 16-bit register which allows resources to have different processor addresses and local peripheral addresses. This register is used along with the PTSR(1-4) register. Bits in this register that have a corresponding bit set in the PTSR(1-4) are driven to the local peripheral bus during a processor bus to local peripheral bus access. These bits correspond to the upper 16 bits of the address bus.

Processor Translation Select Register - PTSR(1-4)

REG	Processor Translation Select Register (1-4)		
ADR/SIZ	PTSR1 - $FFF0001A (16 bits)		
ADR/SIZ	PTSR2 - $FFF0001E (16 bits)		
ADR/SIZ	PTSR3 - $FFF00022 (16 bits)		
ADR/SIZ	PTSR4 - $FFF00026 (16 bits)		
BIT	15	--	0
FIELD	PTSR(1-4)		
OPER	R/W		
RESET	0 P		

The PTSR(1-4) is a 16-bit register that selects which bits of the PTR(1-4), if any, will be driven to the local peripheral bus when an access from the processor bus is decoded. Bits that are set select the corresponding bit in the PTR(1-4) to be driven to the local peripheral bus. Bits that are cleared select the corresponding bit of the processor address to be driven to the local peripheral bus.

System Start Address Register - SSAR(1-4)

REG	System Start Address Register (1-4)		
ADR/SIZ	SSAR1 - $FFF00028 (16 bits)		
ADR/SIZ	SSAR2 - $FFF0002C (16 bits)		
ADR/SIZ	SSAR3 - $FFF00030 (16 bits)		
ADR/SIZ	SSAR4 - $FFF00034 (16 bits)		
BIT	15	--	0
FIELD	SSAR(1-4)		
OPER	R/W		
RESET	0 P		

The SSAR(1-4) is a 16-bit read/write register used to indicate the starting address of a particular memory area on the processor bus to be accessed from the local peripheral bus. The value loaded in this register is compared with the upper 16 bits of the incoming local peripheral bus address. A processor bus request is generated if the address falls in the range specified by the SSAR(1-4) and its corresponding SEAR(1-4) register. In case of overlapping ranges, a priority scheme selects which decoder will be used, with SSAR1/SEAR1 having the highest priority and SSAR4/SEAR4 having the lowest.

System End Address Register - SEAR(1-4)

REG	System End Address Register (1-4)		
ADR/SIZ	SEAR1 - $FFF0002A (16 bits)		
ADR/SIZ	SEAR2 - $FFF0002E (16 bits)		
ADR/SIZ	SEAR3 - $FFF00032 (16 bits)		
ADR/SIZ	SEAR4 - $FFF00036 (16 bits)		
BIT	15	--	0
FIELD	SEAR(1-4)		
OPER	R/W		
RESET	0 P		

The SEAR(1-4) is a 16-bit register used to indicate the ending address of a particular memory area on the local peripheral bus to be accessed from the processor bus. The value loaded in this register is compared with the upper 16 bits of the incoming local peripheral bus address. A processor bus request is generated if the address falls in the range specified by the SSAR(1-4) and its corresponding SEAR(1-4) register. In case of overlapping ranges, a priority scheme selects which decoder will be used, with SSAR1/SEAR1 having the highest priority and SSAR4/SEAR4 having the lowest.

System Translation Register - STR(1-4)

REG	System Translation Register (1-4)		
ADR/SIZ	STR1 - $FFF00038 (16 bits)		
ADR/SIZ	STR2 - $FFF0003C (16 bits)		
ADR/SIZ	STR3 - $FFF00040 (16 bits)		
ADR/SIZ	STR4 - $FFF00044 (16 bits)		
BIT	15	--	0
FIELD	STR(1-4)		
OPER	R/W		
RESET	0 P		

The STR(1-4) is a 16-bit register which allows resources to have different processor addresses and local peripheral addresses. This register is used along with the STSR(1-4) register. Bits in this register that have a corresponding bit set in the STSR(1-4) are driven to the processor bus during a local peripheral bus to processor bus access. These bits correspond to the upper 16 bits of the address bus.

System Translation Select Register - STSR(1-4)

REG	System Translation Select Register (1-4)		
ADR/SIZ	STSR1 - $FFF0003A (16 bits)		
ADR/SIZ	STSR2 - $FFF0003E (16 bits)		
ADR/SIZ	STSR3 - $FFF00042 (16 bits)		
ADR/SIZ	STSR4 - $FFF00046 (16 bits)		
BIT	15	--	0
FIELD	STSR(1-4)		
OPER	R/W		
RESET	0 P		

The STSR(1-4) is a 16-bit register that selects which bits of the STR(1-4), if any, will be driven to the processor bus when an access from the local peripheral bus is decoded. Bits that are set select the corresponding bit in the STR(1-4) to be driven to the processor bus. Bits that are cleared select the corresponding bit of the local peripheral address to be driven to the processor bus.

Processor Attribute Register - PAR(1-4)

REG	Processor Attribute Register (1-4)							
ADR/SIZ	PAR1 - $FFF00048 (8 bits)							
ADR/SIZ	PAR2 - $FFF00049 (8 bits)							
ADR/SIZ	PAR3 - $FFF0004A (8 bits)							
ADR/SIZ	PAR4 - $FFF0004B (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD							WPEN	DEN
OPER	R						R/W	R/W
RESET	0						0	0

This register specifies the attributes to be used during a processor bus to local peripheral bus transfer. Each processor decoder has a corresponding set of attributes. Presently, the attribute bits are:

DEN Decode Enable. If set, the corresponding processor bus decoder is enabled.

WPEN Write Post Enable. If set, write posting is enabled for the corresponding processor bus decoder.

System Attribute Register - SAR(1-4)

REG	System Attribute Register (1-4)							
ADR/SIZ	SAR1 - $FFF0004C (8 bits)							
ADR/SIZ	SAR2 - $FFF0004D (8 bits)							
ADR/SIZ	SAR3 - $FFF0004E (8 bits)							
ADR/SIZ	SAR4 - $FFF0004F (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD					GBL	INVR		DEN
OPER	R				R/W	R/W	R	R/W
RESET	0				0	0	0	0

This register specifies the attributes to be used during a local peripheral bus to processor bus transfer. Each local peripheral bus decoder has a corresponding set of attributes.

DEN Decode Enable. If set, the corresponding local peripheral bus decoder is enabled.

INVR Invalidate on Reads. If set, the INV* signal is asserted on the processor bus during BusSwitch initiated read transfers. Note that the INV* signal is always asserted during write transfers.

GBL Global Access. If set, the GBL* signal is asserted on the processor bus during transfers. This bit must be set to enable bus snooping on the processor bus.

Bus Timer Register - BTIMER

REG	Bus Timer Register							
ADR/SIZ	$FFF00051 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD					SBT		PBT	
OPER	R				R/W		R/W	
RESET	0				0		0	

The BTIMER controls various timers used for terminating bus accesses.

PBT Processor Bus Timeout. This field specifies the processor bus timeout value. The timer starts when TS* is asserted on the processor bus. If TA*, TEA*, or TRTRY* is not asserted before the timer times out, the BusSwitch will assert TEA* on the processor bus. The timer will be disabled if the access goes to the local peripheral bus.

0	8 µsec
1	64 µsec
2	256 µsec
3	Disabled

SBT System Bus Timeout. This field specifies the local peripheral bus timeout value. The timer starts when TS* is asserted on the local peripheral bus. If TA* or TEA* is not asserted before the timer times out, the BusSwitch will assert TEA* on the local peripheral bus. If the cycle was write posted, a write post bus error interrupt is sent to the processor bus. In this case the error address is in the WPA register.

0	8 µsec (default)
1	64 µsec
2	256 µsec
3	Disabled

Prescaler Adjust Register - PADJUST

REG	Prescaler Adjust Register		
ADR/SIZ	$FFF00052 (8 bits)		
BIT	7	--	0
FIELD	PADJUST		
OPER	R/W		
RESET	$CE		

This register is used for specifying a scale factor for the prescaler to ensure that the time base frequency for the tick timers is 1MHz. The scale factor is calculated as follows:

PADJUST = 256 - CLK(MHz), where CLK is the processor clock frequency in megahertzs.

Frequency	PADJUST
50	$CE
40	$D8
33	$DF
25	$E7

Prescaler Count Register - PCOUNT

REG	Prescaler Count Register		
ADR/SIZ	$FFF00053 (8 bits)		
BIT	7	--	0
FIELD	PCOUNT		
OPER	R		
RESET	0		

This register is an 8-bit counter used to generate the 1MHz clock for all the timers in the BusSwitch. This register increments to $FF at the BCLK frequency, then it is loaded from the PADJUST register.

Processor Address Log Register - PAL

REG	Processor Address Log Register		
ADR/SIZ	$FFF00054 (32 bits)		
BIT	31	--	0
FIELD	PAL		
OPER	R		
RESET	0		

The PAL register is a 32-bit read only register that contains the address of the last memory error as signaled by the PALOG input pin. This register is normally updated with each transfer on the processor bus that is not decoded by the BusSwitch. The assertion of PALOG from the ECDM's ERLOG* signal, latches the last bus address and prevents further updates to this register. Refer to the *ECDM Error Logging* section in the *ECDM* chapter.

Write Post Processor Address Register - WPPA

REG	Write Post Processor Address Register		
ADR/SIZ	$FFF00058 (32 bits)		
BIT	31	--	0
FIELD	WPPA		
OPER	R		
RESET	0		

The write post processor address register contains the address of the last write posted cycle. If a write post error interrupt is received, the offending address can be read from this register.

Write Post Translated Processor Address Register - WPTPA

REG	Write Post Translated Processor Address Register		
ADR/SIZ	$FFF0005C (16 bits)		
BIT	15	--	0
FIELD	WPTPA		
OPER	R		
RESET	0		

The write post translated processor address register contains the upper bits of the translated address of the last write posted cycle. If a write post error interrupt is received, the offending address can be read from this register.

Write Post Processor Attributes Register - WPPAT

REG	Write Post Processor Attributes Register							
ADR/SIZ	$FFF0005E (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD		TBST	TSIZ		TC			
OPER	R	R	R		R			
RESET	0	0	0		0			

This register contains the transfer attributes used on the last write posted cycle.

TC Transfer Code.

TSIZ Transfer Size.

TBST Transfer Burst. Set if a burst occurred. Note that this is opposite to the polarity of the MC88110 signal *tbst_*.

ROM Control Register - ROMCR

REG	ROM Control Register							
ADR/SIZ	$FFF00060 (8 bits of 16) (High Byte)							
BIT	15	14	13	12	11	10	0	8
FIELD	ROM0					SGLB	WEN1	WEN0
OPER	R/W	R				R/W	R/W	R/W
RESET	1	0				1	0	0

REG	ROM Control Register							
ADR/SIZ	$FFF00061 (8 bits of 16) (Low Byte)							
BIT	7	6	5	4	3	2	1	0
FIELD	SIZE		ATIME1			ATIME0		
OPER	R/W		R/W			R/W		
RESET	0		7			7		

The BusSwitch provides decoding and control for two banks (0 and 1) of ROMs. This register is used to configure various options for ROM/Flash Memory use.

ATIME0 The ATIME0 field defines the access time of the bank 0 ROMs.

ATIME1 The ATIME1 field defines the access time of the bank 1 ROMs.

The ROM access time is calculated as follows:

$$\text{ROM access}_{nsec} = T(ATIME\mathit{i} + 1) + 5\text{, where}$$

T is the bus clock (BCLK) period in nanoseconds. ATIME*i* is the value in the ATIME0 or ATIME1 fields.

SIZE The SIZE field defines the size of the ROM space. The ending address of bank 0 and the starting address of bank 1 are specified by this field:

SIZE	ROM Size	Bank 0	Bank 1
0	8Mbit (512K x 16)	FF800000-FF9FFFFF	FFA00000-FFBFFFFF
1	4Mbit (256K x 16)	FF800000-FF8FFFFF	FF900000-FF9FFFFF
2	2Mbit (128K x 16)	FF800000-FF87FFFF	FF880000-FF8FFFFF
3	1Mbit (64K x 16)	FF800000-FF83FFFF	FF840000-FF87FFFF

WEN0 Write Enable 0. This bit is used for flash memory. If set, and the XCC bit in the GCSR is cleared, write transfers to the bank 0 ROM memory space are enabled. In this case, the MEMWEN* signal is asserted during the transfer.

WEN1 Write Enable 1. This bit is used for flash memory. If set, and the XCC bit in the GCSR is cleared, write transfers to the bank 1 ROM memory space are enabled. In this case, the MEMWEN* signal is asserted during the transfer.

SGLB Single Bank. If set, this bit indicates that only one bank is used, with the size of the bank twice the size specified in the SIZE field.

ROM0 ROM at 0 enable. This bit controls the external ROM decoder. If set, decoding will be enabled for the lower 4 megabytes of the address space starting at $00000000 and a MC68040 bus cycle will be generated for any access in this space. If ROM0 is cleared, the external ROM decoder will be disabled. Note that after reset, this bit is set, providing a mechanism for fetching the reset vector.

Timer Control 1 Register - TCTRL1

REG	Timer Control 1 Register							
ADR/SIZ	$FFF00062 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	OVF					COVF	COC	CEN
OPER	R				R	C	R/W	R/W
RESET	0				0	0	0	0

The TCTRL1 register controls operation of timer 1.

CEN Counter Enable. If set, the counter is enabled and increments by 1 every microsecond.

COC Clear on Compare. If set, the counter is cleared when the counter value is equal to the contents of the timer compare 1 register.

COVF Clear Overflow Counter. The overflow counter is cleared when a one is written to this bit.

OVF Overflow Counter. The overflow counter is incremented when the counter value is equal to the contents of the timer compare 1 register.

Timer Control 2 Register - TCTRL2

REG	Timer Control 2 Register							
ADR/SIZ	$FFF00063 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	OVF					COVF	COC	CEN
OPER	R				R	C	R/W	R/W
RESET	0				0	0	0	0

The TCTRL2 register controls operation of timer 2.

CEN Counter Enable. If set, the counter is enabled and increments by 1 every microsecond.

COC Clear on Compare. If set, the counter is cleared when the counter value is equal to the contents of the timer compare 2 register.

COVF Clear Overflow Counter. The overflow counter is cleared when a one is written to this bit.

OVF Overflow Counter. The overflow counter is incremented when the counter value is equal to the contents of the timer compare 2 register.

Interrupt Level Register - LEVEL

REG	Interrupt Level Register							
ADR/SIZ	$FFF00064 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD						LEVEL		
OPER	R					R		
RESET	0					0		

The LEVEL Register contains the priority encoded interrupt level. This level is a combination of the BusSwitch interrupt requests, the interrupts driven onto the IPL lines, and the interrupts driven onto the XIPL lines (if enabled). (The MVME197 module series need to enable the XIPL lines to receive interrupt requests from the VMEchip2). LEVEL is continuously updated to reflect the highest priority interrupt received. The highest interrupt level is latched in the LEVEL register. A write to the level register (any value) clears this latch, allowing lower level interrupts to be received. Note that clearing the interrupt source or changing the ISEL0/ISLE1 registers can cause the LEVEL register to be updated.

LEVEL This field contains the interrupt level.

Interrupt Mask Register - MASK

<table>
<tr><td><b>REG</b></td><td colspan="8">Interrupt Mask Register</td></tr>
<tr><td><b>ADR/SIZ</b></td><td colspan="8">$FFF00065 (8 bits)</td></tr>
<tr><td><b>BIT</b></td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td><b>FIELD</b></td><td colspan="5"></td><td colspan="3">MASK</td></tr>
<tr><td><b>OPER</b></td><td colspan="5">R</td><td colspan="3">R/W</td></tr>
<tr><td><b>RESET</b></td><td colspan="5">0</td><td colspan="3">0</td></tr>
</table>

The MASK Register contains the encoded interrupt priority mask. An interrupt received at the same or lower level than the one in the MASK register will be ignored. Interrupts received above the mask level will generate an interrupt signal to the processor.

MASK This field contains the interrupt mask.

Interrupt Select 0 Register - ISEL0

<table>
<tr><td><b>REG</b></td><td colspan="8">Interrupt Select 0 Register</td></tr>
<tr><td><b>ADR/SIZ</b></td><td colspan="8">$FFF00066 (8 bits)</td></tr>
<tr><td><b>BIT</b></td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td><b>FIELD</b></td><td>SEL7</td><td>SEL6</td><td>SEL5</td><td>SEL4</td><td>SEL3</td><td>SEL2</td><td>SEL1</td><td></td></tr>
<tr><td><b>OPER</b></td><td>S</td><td>S</td><td>S</td><td>S</td><td>S</td><td>S</td><td>S</td><td>R</td></tr>
<tr><td><b>RESET</b></td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr>
</table>

(This register is applicable only to the MVME197DP modules of the MVME197 single board computer series).

This 8-bit register selects those interrupts that will be handled by processor 0. Bit positions 1 through 7 correspond with interrupt levels 1-7. Writing a 1 to any SEL*i* bit will set the bit and will clear the corresponding SEL*i* bit in the ISEL1 register. This will route all the interrupts at the corresponding level to processor 0. Note that writing a 0 will have no effect on any of these bits.

Interrupt Select 1 Register - ISEL1

REG	Interrupt Select 1 Register							
ADR/SIZ	$FFF00067 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	SEL7	SEL6	SEL5	SEL4	SEL3	SEL2	SEL1	
OPER	S	S	S	S	S	S	S	R
RESET	0	0	0	0	0	0	0	0

(This register is applicable only to the MVME197DP modules of the MVME197 single board computer series).

This 8-bit register selects those interrupts that will be handled by processor 1. Bit positions 1 through 7 correspond with interrupt levels 1-7. Writing a 1 to any SEL*i* bit will set the bit and will clear the corresponding SEL*i* bit in the ISEL0 register. This will route all the interrupts at the corresponding level to processor 1. Note that writing a 0 will have no effect on any of these bits.

Abort Control Register - ABORT

REG	Abort Control Register							
ADR/SIZ	$FFF00068 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD		ABT	INT	IEN	ICLR			
OPER	R	R	R	R/W	C	R		
RESET	0	0	0	0	0	0		

This register controls operation of the ABORT interrupt. The abort interrupt is only received by CPU0.

ICLR Interrupt Clear. Writing a 1 to this bit clears the INT status bit. This bit always reads as 0.

IEN Interrupt Enable. If set, interrupts are enabled. If cleared, interrupts are disabled.

INT Interrupt. This bit is set if the ABORT interrupt is received.

ABT Abort Signal. This bit reflects the state of the abort pin. If set, the abort signal is asserted.

Cross Processor Interrupt Register - CPINT

REG	Cross Processor Interrupt Register							
ADR/SIZ	$FFF00069 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	SCPI	CINT	INT	IEN	ICLR			
OPER	S	R	R	R/W	C	R		
RESET	0	0	0	0	0	0		

(This register is applicable only to the MVME197DP modules of the MVME197 single board computer series).

This register controls operation of the cross processor interrupt.

ICLR Interrupt Clear. Writing a 1 to this bit clears the INT status bit. This bit always reads as 0.

IEN Interrupt Enable. If set, interrupts are enabled. If cleared, interrupts are disabled.

INT Interrupt. This bit is set if the CPINT interrupt is received.

CINT Cross Interrupt. This bit reflects the INT bit of the CPINT register of the other processor.

SCPI Send Cross Processor Interrupt. Setting this bit sends a cross processor interrupt to the other processor. This bit always reads as 0.

Timer Interrupt 1 Register - TINT1

REG	Timer Interrupt 1 Register							
ADR/SIZ	$FFF0006A (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD			INT	IEN	ICLR	ILVL		
OPER	R		R	R/W	C	R/W		
RESET	0		0	0	0	0		

The TINT1 register controls interrupt operation for timer 1.

ILVL Interrupt Level. This field selects the interrupt level for timer 1. Level 0 does not generate an interrupt.

ICLR Interrupt Clear. Writing a 1 to this bit clears the INT status bit. This bit always reads as 0.

IEN Interrupt Enable. If set, interrupts are enabled. If cleared, interrupts are disabled.

INT Interrupt Status. This bit is set when an interrupt is generated by timer 1.

Timer Interrupt 2 Register - TINT2

REG	Timer Interrupt 2 Register							
ADR/SIZ	$FFF0006B (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD			INT	IEN	ICLR	ILVL		
OPER	R		R	R/W	C	R/W		
RESET	0		0	0	0	0		

The TINT2 register controls interrupt operation for timer 2.

ILVL Interrupt Level. This field selects the interrupt level for timer 2. Level 0 does not generate an interrupt.

ICLR Interrupt Clear. Writing a 1 to this bit clears the INT status bit. This bit always reads as 0.

IEN Interrupt Enable. If set, interrupts are enabled. If cleared, interrupts are disabled.

INT Interrupt Status. This bit is set when an interrupt is generated by timer 2.

Write Post Interrupt Control Register - WPINT

REG	Write Post Interrupt Control Register							
ADR/SIZ	$FFF0006C (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD			INT	IEN	ICLR	LVL		
OPER	R		R	R/W	C	R/W		
RESET	0		0	0	0	0		

This register controls operation of the write post interrupt.

LVL Interrupt Level.

ICLR Interrupt Clear. Writing a 1 to this bit clears the INT status bit. This bit always reads as 0.

IEN Interrupt Enable. If set, interrupts are enabled. If cleared, interrupts are disabled.

INT Interrupt. This bit is set if the WPINT interrupt is received.

Processor Address Log Interrupt Register - PALINT

REG	Processor Address Log Interrupt Register							
ADR/SIZ	$FFF0006D (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	PLTY		INT	IEN	ICLR	LVL		
OPER	R/W	R	R	R/W	C	R/W		
RESET	0	0	0	0	0	0		

This register controls operation of the processor address log interrupt. The ECDM asserts the ERLOG * signal to the BusSwitch PALINT* pin, which causes the last processor address to be latched in the PAL register.

LVL Interrupt Level.

ICLR Interrupt Clear. Writing a 1 to this bit clears the INT status bit. This bit always reads as 0.

IEN Interrupt Enable. If set, interrupts are enabled. If cleared, interrupts are disabled.

INT Interrupt. This bit is set if the PALINT* signal is asserted, and the IEN bit is set.

PLTY Polarity. If set, an interrupt is detected on the rising edge of the IRQ* pin. If cleared, an interrupt is detected on the falling edge of the IRQ* pin.

External Interrupt Register- XINT

REG	External Interrupt Register							
ADR/SIZ	$FFF0006E (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	PLTY	E/L*	INT	IEN	ICLR	LVL		
OPER	R/W	R/W	R	R/W	C	R/W		
RESET	0	0	0	0	0	0		

This register controls operation of the external interrupt request signal, IRQ*. (There is no source on the MVME197LE versions of the MVME197 series that generates this IRQ* signal). This pin is also connected to the mezzanine connector, which could be a potential source for this IRQ* if the mezzanine board is able to generate IRQ* to this pin.

LVL Interrupt Level.

ICLR Interrupt Clear. Writing a 1 to this bit clears the INT status bit. This bit always reads as 0.

IEN Interrupt Enable. If set, interrupts are enabled. If cleared, interrupts are disabled.

INT Interrupt. This bit is set if the IRQ* interrupt is received.

E/L* Edge/Level. If set, the interrupt is edge sensitive (latched). If cleared, the interrupt is level sensitive.

PLTY Polarity. If set, an interrupt is detected on the rising edge or on a high level on the IRQ* pin (as selected by the E/L* bit). If cleared, an interrupt is detected on the falling edge or on a low level on the IRQ* pin (as selected by the E/L* bit).

Vector Base Register - VBASE

REG	Vector Base Register							
ADR/SIZ	$FFF0006F (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	BASE					SRC		
OPER	R/W					R		
RESET	0				1	1	1	1

The VBASE is an 8-bit read/write register used to supply the vector to the CPU during the interrupt acknowledge cycle for the BusSwitch generated interrupts. After reset, this register contains the value $0F, and this will be the value returned during vector read cycles until VBASE is initialized. During interrupts, one of the VECTOR registers is read to obtain the interrupt vector.

SRC Source. This field is a place holder for the interrupt source when reading the vector for internally generated interrupts. It will read as zeros once BASE is initialized.

BASE Base. The value written to this field is returned as bits 7:3 of the vector for internally generated interrupts.

The interrupt source encodings are as follows:

0	Timer Interrupt 1.
1	Timer Interrupt 2.
2	Write Post Error Interrupt.
3	Processor Address Log Interrupt.
4	External Interrupt.
7	Spurious Interrupt.

Timer Compare 1 Register - TCOMP1

REG	Timer Compare 1 Register		
ADR/SIZ	$FFF00070 (32 bits)		
BIT	31	------------------------------------	0
FIELD	TCOMP1		
OPER	R/W		
RESET	0		

The TCOMP1 register is compared to the timer 1 counter, and if they are equal, the overflow counter is incremented, and an interrupt is generated (if enabled). If the clear on compare mode is enabled in the TCTRL1 register, the counter is cleared.

Timer Counter 1 Register - TCOUNT1

REG	Timer Counter 1 Register		
ADR/SIZ	$FFF00074 (32 bits)		
BIT	31	------------------------------------	0
FIELD	TCOUNT1		
OPER	R/W		
RESET	0		

The TCOUNT1 is the counter for timer 1. Its operation is controlled by various bits in the TCTRL1 register. If enabled, it increments its value every microsecond, and can be read or written at any time. This register is compared with TCOMP1, and if they are equal, the overflow counter is incremented, and an interrupt is generated if enabled. If the clear-on-compare mode is enabled, the counter is cleared.

Timer Compare 2 Register - TCOMP2

REG	Timer Compare 2 Register		
ADR/SIZ	$FFF00078 (32 bits)		
BIT	31	--	0
FIELD	TCOMP2		
OPER	R/W		
RESET	0		

The TCOMP2 register is compared to the timer 2 counter, and if they are equal, the overflow counter is incremented, and an interrupt is generated (if enabled). If the clear on compare mode is enabled in the TCTRL2 register, the counter is cleared.

Timer Counter 2 Register - TCOUNT2

REG	Timer Counter 2 Register		
ADR/SIZ	$FFF0007C (32 bits)		
BIT	31	--	0
FIELD	TCOUNT2		
OPER	R/W		
RESET	0		

The TCOUNT2 is the counter for timer 1. Its operation is controlled by various bits in the TCTRL2 register. If enabled, it increments its value every microsecond, and can be read or written at any time. This register is compared with TCOMP2, and if they are equal, the overflow counter is incremented, and an interrupt is generated if enabled. If the clear-on-compare mode is enabled, the counter is cleared.

General Purpose Register - GPR(1-4)

REG	General Purpose Register (1-4)		
ADR/SIZ	GPR1 - $FFF00080 (32 bits)		
ADR/SIZ	GPR2 - $FFF00084 (32 bits)		
ADR/SIZ	GPR3 - $FFF00088 (32 bits)		
ADR/SIZ	GPR4 - $FFF0008C (32 bits)		
BIT	31	--	0
FIELD	GPR(1-4)		
OPER	R/W (32 bits)		
RESET	0 P		

Four 32-bit general purpose read/write registers are provided for storage. These registers do not control any hardware.

External Cache TAGS Register - XCTAGS

REG	External Cache TAGS Register		
ADR/SIZ	$FFF00090 (32 bits)		
BIT	31	--	0
FIELD	XCTAGS		
OPER	R		
RESET	0		

(This register is applicable only to the MVME197DP/SP versions of the MVME197 series).

This register is used to capture the MC88410 tags during diagnostic system invalidate cycles.

External Cache Control Register - XCCR

REG	External Cache Control Register		
ADR/SIZ	$FFF00100 (24 bits)		
BIT	31	--	8
FIELD			
OPER	R		
RESET	$FFF		

REG	External Cache Control Register							
ADR/SIZ	$FFF00103 (8 bits of 32)(Low Byte)							
BIT	7	6	5	4	3	2	1	0
FIELD					DIAG	FBSY	F1	F0
OPER	R				R/W	R	R	R
RESET	$F				0	0	0	0

(This register is applicable only to the MVME197DP/SP versions of the MVME197 series). This register provides control and status information for the MC88410 cache controller. Only the lower 4 bits are implemented. To access this register, the XCC bit in the BusSwitch GCSR register must be set.

F1-F0 Flush Control Bits. These bits specify the flush operation sent to the MC88410 while FBSY is 0. They will be cleared when FBSY is set.

F1	F0	Operation
0	0	No operation
0	1	Flush page
1	0	Flush all
1	1	Invalidate all

FBSY Flush Busy. This bit will be set while a flush or invalidate operation is in progress. After initiating a flush/invalidate operation, software should poll FBSY and wait until is set before proceding.

DIAG Diagnostic Mode. Setting this bit will put the corresponding MC88410 in diagnostic mode. All subsequent transactions will be interpreted as diagnostic accesses.

Vector Registers - VECTOR(1-7)

REG	Vector Registers (1-7)		
ADR/SIZ	Vector1 - $FFF00104 (24 bits)		
ADR/SIZ	Vector2 - $FFF00108 (24 bits)		
ADR/SIZ	Vector3 - $FFF0010C (24 bits)		
ADR/SIZ	Vector4 - $FFF00110 (24 bits)		
ADR/SIZ	Vector5 - $FFF00114 (24 bits)		
ADR/SIZ	Vector6 - $FFF00118 (24 bits)		
ADR/SIZ	Vector7 - $FFF0011C (24 bits)		
BIT	31	------------------------------------	8
FIELD	VECTOR(1-7)(Upper 3 Bytes)		
OPER	R		
RESET	0		

REG	Vector Registers (1-7)							
ADR/SIZ	Vector1 - $FFF00107 (8 bits)(Low Byte)							
ADR/SIZ	Vector2 - $FFF0010B (8 bits)(Low Byte)							
ADR/SIZ	Vector3 - $FFF0010F (8 bits)(Low Byte)							
ADR/SIZ	Vector4 - $FFF00113 (8 bits)(Low Byte)							
ADR/SIZ	Vector5 - $FFF00117 (8 bits)(Low Byte)							
ADR/SIZ	Vector6 - $FFF0011B (8 bits)(Low Byte)							
ADR/SIZ	Vector7 - $FFF0011F (8 bits)(Low Byte)							
BIT	7	6	5	4	3	2	1	0
FIELD	VECTOR(1-7)(Low Byte)							
OPER	R							
RESET	$0 F							

There are 7 vector registers, one for each interrupt level, with VECTOR*i* corresponding to level *i* (1 to 7). Based on the interrupt level read from the LEVEL register. The VECTOR read should correspond with the value read from the LEVEL register. If the BusSwitch is generating an interrupt at that level, the vector read will be generated by the BusSwitch as specified by the VBASE register. If the BusSwitch is not generating an interrupt at that level, an IACK cycle will be generated on the local peripheral bus to obtain the vector.

External Cache Flush Register - XCFR

REG	External Cache Flush Register		
ADR/SIZ	$FF8PPPPP (24 bits)		
BIT	63	--	40
FIELD			
OPER	W		

REG	External Cache Flush Register							
ADR/SIZ	$FF8PPPPP (8 bits of 32)							
BIT	39	38	37	36	35	34	33	32
FIELD							F1	F0
OPER	W						W	W

REG	External Cache Flush Register		
ADR/SIZ	$FF8PPPPP (32 bits)		
BIT	31	--	0
FIELD			
OPER	W		

(This register is applicable only to the MVME197DP/SP versions of the MVME197 series).

This register provides flushing and invalidation control for the MC88410 cache controller. To access this register, the XCC bit in the BusSwitch GCSR register must be set and the WEN1 and WEN0 bits in the BusSwitch ROMCR register must be cleared. Also, the misaligned exception mask (MXM) bit in the MC88110 processor status register (cr1) must be set. Only a double write (st.d) transfer should be used to access this register.

PPPPP — This field represents the lower 20 bits of the address. It should be set to the page address for a flush page operation, and to 0 for anything else.

F1-F0 — Flush Control Bits. These bits specify the flush operation to the MC88410 cache controller. They are cleared by hardware after the FBSY bit is set in the XCCR register.

F1	F0	Operation
0	0	No operation
0	1	Flush page
1	0	Flush all
1	1	Invalidate all

Bus Operation

The BusSwitch provides a master and a slave interface to the processor bus and to the local peripheral bus. The processor bus follows the MC88110 bus protocol, and the local peripheral bus follows the MC68040 bus protocol. MC88110 cycles will be translated to MC68040 type cycles and vice-versa as shown below.

Table 4-2. MC88110 Transfer Code Translation

MC88110 Transfer Code (TC)	MC68040 Transfer Modifier (TM) Transfer Type (TT) = 00	Description
0000	111	Reserved
0001	001	User Data
0010	001	User Touch/Flush/Allocate
0011	011	MMU Data
0100	000	Replacement Copyback
0101	101	Supervisor Data
0110	101	Supervisor Touch/Flush/Allocate
0111	000	Snoop Copyback
1000	111	Reserved
1001	010	User Code
1010	111	Reserved
1011	100	MMU Code
1100	111	Reserved
1101	110	Supervisor Code
1110	111	Reserved
1111	111	Reserved

Table 4-3. MC68040 Transfer Modifier Translation

MC68040 Transfer Modifier (TM) Transfer Type (TT) = 00	MC88110 Transfer Code (TC)	Description
000	0111	Data Cache Push
001	0001	User Data
010	1001	User Code
011	0011	MMU Data
100	1011	MMU Code
101	0101	Supervisor Data
110	1101	Supervisor Code
111	1111	Reserved

Processor Slave Interface

The slave interface is used by the current bus master to access internal registers of the BusSwitch or for access to resources on the local peripheral bus. For access to onboard registers, single beat transfers only are allowed. These cycles will be terminated with TA*. Burst transfers to processor registers will be terminated with TEA*. This means that all BusSwitch register accesses should be cache inhibited. An access to a BusSwitch register will always be terminated with TA* or TEA*, and will never be pre-empted by a retry cycle. If the access is to the local peripheral bus, a system bus request will be generated. After receiving a system bus grant, one or more cycles will be run on the local peripheral bus. An access to the local peripheral bus can be pre-empted by a bus retry operation. This occurs if a local peripheral bus master has started a cycle to access a resource on the processor bus while the processor bus master is trying to access the local peripheral bus. The BusSwitch will send a retry signal to the current processor bus master, it will request the processor bus, and after receiving a bus grant it will run the cycle requested from the local peripheral bus. Once finished, it will release the processor bus, allowing the pre-empted master to retry its cycle.

Processor Master Interface

The Master Interface is used by the BusSwitch to access the processor bus on behalf of the current local peripheral bus master. The BusSwitch can be pre-empted during transfers if a snoop hit is signaled by the MC88110. This can happen if the GBL bit is set in the corresponding system attribute register and the entry hits in the MC88110 data cache. In this case the BusSwitch will relinquish the bus. The snooping CPU will arbitrate and obtain processor bus mastership, and then it will update memory. The BusSwitch will then request the processor bus and once it becomes the bus master it will retry the aborted cycle.

Dual Processor Registers

For proper operation with two processors, some registers are actually implemented as two registers (one for each processor) mapped to the same address. The BusSwitch can distinguish which processor is doing the access and select the corresponding register. The following registers are duplicated:

LEVEL

MASK

CPINT

The CPUID bit in the GCSR can be read by a processor to find its ID (0 or 1). (The MVME197LE/SP versions of the MVME197 series should always have ID = 0).

DRAM CONTROLLER AND ADDRESS MULTIPLEXER (DCAM) 5

Introduction

This chapter describes the architecture and usage of the DRAM Controller and Address Multiplexer gate array, otherwise referred to as the DCAM. The DCAM provides all the control and timing required to interface to a dynamic RAM array from the MC88110 processor bus.

DCAM Features

These are some of the features of the DCAM gate array:

- ❑ Allows one or two banks of DRAM - each consisting of 72, x 4, DRAMs
- ❑ Uses Static Column DRAMs - allowing faster consecutive accesses
- ❑ Supports 1M x 4 and 4M x 4 devices
- ❑ Two lines of cache data (the current 256 bits and the next consecutive 256 bits)
- ❑ Programmability to accommodate 33MHz - 66MHz bus operation
- ❑ Programmable soft error scrubbing
- ❑ Refresh selectable from internal counter or external input allows staggered refresh
- ❑ Low power sleep mode
- ❑ Slow mode to throttle system in battery powered mode
- ❑ Allows use of Page Mode DRAMs without look-ahead feature

System Overview

The DCAM, along with the Error Correction and Data Multiplexer (ECDM) provides the interface required for connecting the MC88110 bus to the local Dynamic RAM. A simplified block diagram of the MVME197 and how the DCAM is implemented on that single board computer is shown in Figure 5-1.

This system is designed to use Static Column DRAMs. This type DRAM latches the RAS addresses on the RAS signal and uses the currently applied CAS addresses after the CAS signal goes low. If the CAS addresses change, at the CAS-to-data time later, the data pointed to by the new CAS address is presented at the data out pins. This feature can be used to quickly get to any data location within a given RAS address range.

An alternative method is provided to use the standard Page Mode DRAMs in the event Static Column DRAM availability is a problem. On power up, the chip is configured for Page Mode DRAMs.

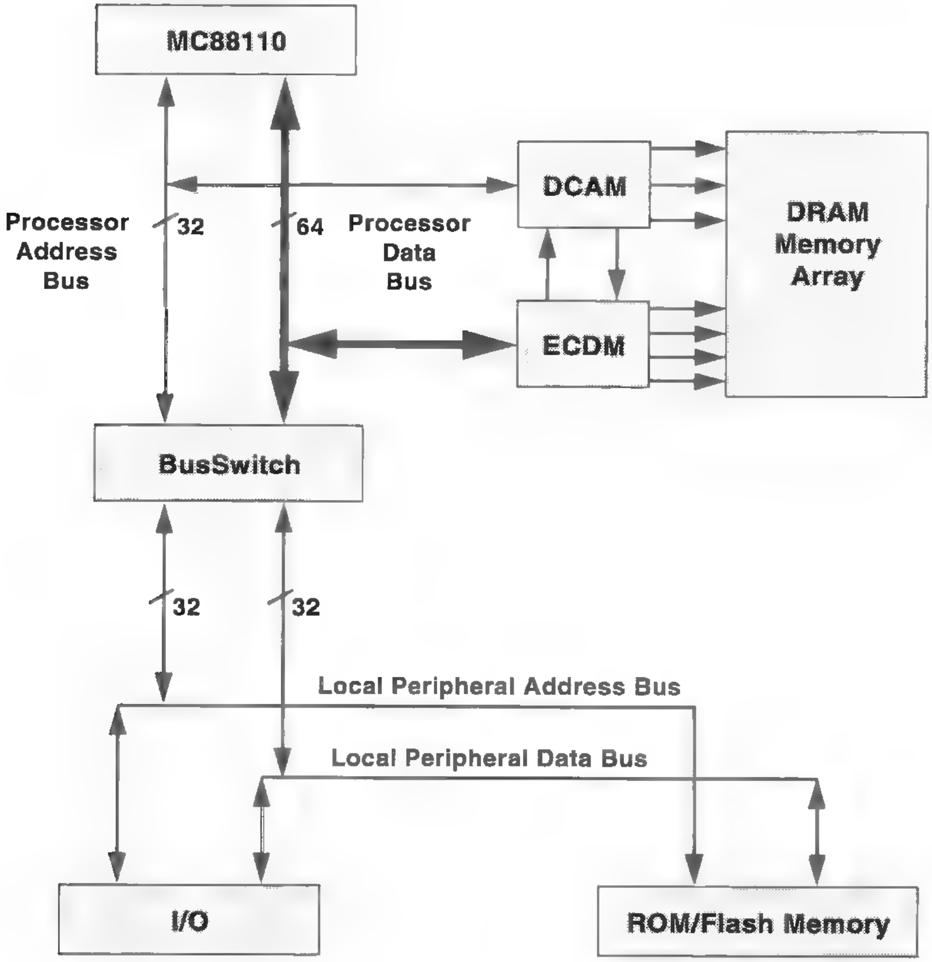


Figure 5-1. Simplified MVME197 Block Diagram

DCAM Gate Array Functions

General Operation

The DCAM, along with 4 ECDM chips, is intended to interface between an MC88110 or MC88410 compatible bus and one or two banks of DRAM. The DCAM provides address decoding, DRAM timing and control of the ECDM chips. Many of the hardware controls in the DCAM are programmable to allow compatibility with various clock speeds, DRAM speeds and page mode or static column DRAMs. These are not intended to be used by operating software, but are configured and not changed. There are other controls that are software dependent, including mapping of the DRAM and CSR location in the memory map, scrub control and refresh timing. Both hardware and software controls are reached through the serial I^2C bus (refer to the *ECDM* chapter for the definition of the serial I^2C bus) and reside in the same group of registers. There are three ways control values are established. On power up, default values are loaded into the registers. These are general values which should work for any situation, but are not optimized. After power up, the I^2C bus looks for the serial EEPROM in the I^2C bus. If it is found, it will automatically load the values from the serial EEPROM into the control registers. These should be optimized for the clock speeds and DRAMs installed in the system. If the serial EEPROM is not present, or if it is desired to change the values later, this can be done through the I^2C bus via the ECDM I^2C bus master interface registers.

As reset goes high, the signal PTA is monitored. If it is high, the DCAM is configured for MC88110 operation. If it is low, it is configured for MC88410 operation. This cannot be changed except at the end of a reset pulse.

Once the part is configured, it will monitor the processor bus signals for a match of the processor address lines to the address in the decode registers. If there is a match, it will begin a cycle. It will also compare the address on the processor bus with the address in the cache address tag registers and look at the tags for those registers. If it is a read and the cache tags match, it will immediately output the stored data. Otherwise it will begin a DRAM cycle.

Many of the functions in the DRAM cycle are programmable, including the length of time to the TA signal.

Read Cycles

The steps in a non-cache read cycle would generally be as follows: Processor addresses, TS, and control signals are presented on the bus on the first clock. RAS addresses are passed on through to the DRAMs. On the second clock, PTA is sent to the processor and DTS is sent to the ECDM so that the ECDM can load the current address. RAS is lowered, CAS addresses for the location of the first line are latched into the CAS address latches/counters. DRAM output enable is driven low. A short time later, depending on the times programed in, CAS is driven low. Output of the ECDM to the processor bus is enabled. At the time data would be valid, the DCAM drives the latch signal to latch the first line of data into the ECDM and also advances the CAS address to the location of the second line. If the DCAM is programed for static column DRAMs, a short time later the second line will be latched into the ECDM and the CAS address will again advance, holding the address of the third line. At precharge time, RAS will go high. At the time programed in, TA will be driven to the processor. CAS will go high and, after the time programed into precharge, the DCAM will be ready to accept a new cycle. The cache address tag or tags will hold the addresses of any lines read, and the valid bits for the line or lines will be set. In this example, one set of latches in the ECDM holds the first line of data and the other holds the second line of data. The CAS address latches/counters will hold the CAS address of the third line of data.

Cache Read Cycles

The steps in a cache read cycle would generally be as follows: It will be assumed that this cycle follows the one just described. Processor addresses for the second line of data and TS, and control signals are presented on the bus on the first clock. One of the cache address tags matches the processor addresses for the second line of data and the corresponding valid bit is set. RAS addresses are passed on through to the DRAMs. On the second clock, PTA is sent to the processor and DTS is sent to the ECDM so that the ECDM can load the current address. If it is a burst, RAS is driven low, if it is not a burst, no DRAM access is done. The output enable from the ECDM to the processor is turned on and the second line of data is driven to the processor. On the third clock, TA is driven to the processor. If it is not a burst cycle, TA is taken away before the next clock and output enables are removed and the cycle is over. If it is a burst cycle, the CAS addresses from the CAS address/counters, which are for the third line of data, are presented to the DRAM and CAS is driven low. Shortly after that, the third line of data is latched into the ECDM, the CAS addresses in the latch/counters are advanced to the address of the fourth line of data, and RAS is taken high. After the time programed into the precharge, the cycle is ended. The ECDM will hold the second and third lines of data and the CAS address latches/counters will point to the address of the fourth line of data.

Normal (non-burst) Write Cycles

The steps in a normal write cycle would generally be as follows: Processor addresses and TS, and control signals are presented on the bus on the first clock. RAS addresses are passed on through to the DRAMs. On the second clock, PTA is sent to the processor and DTS is sent to the ECDM so that the ECDM can load the current address. RAS is lowered, CAS addresses for the location of the first line are latched into the CAS address latches/counters. DRAM output enable is driven low. A short time later, depending on the times programed in, CAS is driven low. At the time data would be valid, the DCAM drives the latch signal to latch the line of data into the ECDM. The output enable to the DRAM chips is turned off. The line read is checked for errors, any single-bit errors are corrected and the new byte or bytes are merged with the old data to be retained. The output from the ECDM to the DRAMs is turned on, data is driven to the DRAMs and the write line to the DRAMs is lowered. RAS is taken high, CAS is taken high and when the precharge time is met, a new cycle can begin. All valid bits for data stored in the cache are cleared.

Burst Write Cycles

The steps in a burst write cycle would generally be as follows: Processor addresses and TS, and control signals are presented on the bus on the first clock. RAS addresses are passed on through to the DRAMs. On the second clock, PTA is sent to the processor and DTS is sent to the ECDM so that the ECDM can load the current address. RAS is lowered, CAS addresses for the location of the line to be written are latched into the CAS address latches/counters. The output enable from the ECDM to the DRAMs is turned on. A short time later, depending on the times programed in, CAS is driven low. On the third clock, the first longword is latched into the ECDMs and Transfer Acknowledge is driven to the processor. On each of the next three clocks, a new longword of data is latched into the ECDMs, until the entire line is latched. The write line is driven low, RAS is taken high, then CAS is taken high and after the precharge time is met, a new cycle can begin. All valid bits for data stored in the cache are cleared.

Functional Block Description

Processor Address Buffers

Processor address buffers only buffer the processor addresses from the MC88110 bus. They are latched internally at various places. The processor addresses are also monitored by the JTAG scan cells.

State Machine Registers

The state machine registers are programed by a two-bit serial bus and contain the specialized instructions for programing the DCAM to the specific timing requirements for any application.

Cycle Decode

The cycle decode circuits decode for eight type cycles. The decoded cycle is latched when it is selected by the arbitration and is started. The possible cycles are:

1. Normal read-two lines read, cached, and one longword or less returned to the requesting device when configured for Static Column DRAMs. One line read, cached and one longword or less returned to the requesting device when configured for Page Mode DRAMs.
2. Burst read-two lines read, cached, one line returned to the requesting device when configured for Static Column DRAMs. One line read, cached and one line returned to the requesting device when configured for Page Mode DRAMs.
3. Read modify write (normal write)-one longword or less written, preformed as a read of one line, error check, merge of new byte or bytes, re-writing the line.
4. Burst write-one line stored, entire line written.
5. Cache hit read-one longword or less sent to the requesting device from cache, no DRAM access preformed.
6. Cache burst read-one line sent to the requesting device from cache, one line read from DRAM and stored in cache when configured for Static Column DRAMs. This cycle type is not available when configured for Page Mode DRAMs.
7. Scrub cycle-read modify write cycle preformed on one line with no new data. The old or corrected data is re-written.
8. Refresh cycle-one CAS before RAS refresh preformed.

Address Decode

There are two independent address decode circuits, one for each of the two banks of RAM supported. Each is mapable on boundaries of the size of the bank. Factored into the decode are bits which shows the size of the DRAM chips installed. Decoding is disabled while the inputs "DISABLE" or ROM0 are held low. This allows ROM accesses for vector fetches. If the two banks are mapped the same, the high bank is disabled.

Timing State Machine

The timing state machine consists of two levels. One level consists of arbitration to select the cycle to be run and a register to store a cycle pending, should a new cycle be requested while a previous one is still in progress. The second level selects the various steps to run the specific cycle. Many of the steps are independently programmable, such as the length of RAS, time from RAS to CAS, the time to each strobe and latch, output enables and the time of RAS precharge. Once a cycle is started, the state machine takes each cycle through to the beginning of the next cycle.

CAS Boundary Checker

When the highest CAS address is reached in a RAS address range, new CAS addresses will have to be loaded to allow the next cache look ahead cycle. The CAS boundary checker detects when the CAS counters have reached this CAS boundary. This signals the look-ahead circuits that a normal cycle must be run and the counters must be reloaded.

CAS Latches and Counters

The CAS addresses are held in a set of latches/counters. At the beginning of any cycle, it is determined if the cycle is a cache hit read cycle. If it is not, new addresses are loaded into the counters.

When configured with static column DRAMs, if it is a read, but not a cache hit, as soon as the data from the first locations are latched in one set of registers in the ECDM, the CAS counters are incremented to the next location. The Static Column DRAMs then output the new data and it is latched in the other set of latches in the ECDM. The CAS counters are incremented once more to point to the next consecutive location to be ready for a new cycle in case the next cycle is also a cache hit read.

When configure with page mode DRAMs, the first location is latched into the ECDM. No further action takes place after that.

When a cycle is decoded to be a cache hit burst read, the CAS counters are not loaded, but the current value, which points to one location past the data held in the ECDM registers, is used. As the stored data is sent to the processor from one set of registers in the ECDM, a memory cycle is run to get the next location and the data is stored in the other set of registers in the ECDM. The CAS counters are then advanced to the next location.

If the cycle is decoded to be a write, either a burst write or a longword or less, the CAS addresses are latched into the register/counters at the beginning of the cycle and used for that cycle. They are not incremented or used again.

RAS Address Latches

RAS addresses are latched into the RAS Address Latches at the beginning of each cycle. When valid data is latched in the ECDM, the latched RAS addresses are latched into the address TAG registers, along with the current CAS address. These RAS Address latches are used because the RAS Addresses could be gone from the bus by the time the new data is latched on a Cache read.

Address Multiplexer

The address multiplexers select addresses from either the bus (for RAS) and the CAS latches/counters (for CAS) during memory cycles from the bus, or from the SCRUB counters during scrub cycles.

Cache Address TAGs and Comparators

There are two cache address tag registers with valid bits for the cached lines of data. The cache address tag register is loaded with the current memory address when read data is latched into the ECDM from DRAM. At the same time, the corresponding valid bit is set. If a write or scrub cycle is done, this bit is cleared to indicate the line of data is no longer valid.

When a read cycle is decoded from the bus, this valid bit is checked and the bus address is compared to the cache address tag register to see if the bus address matches either line stored. If these conditions are met, a cache read is performed.

Refresh Counters and Control

Refresh control consists of a set of registers in the control register section which hold a programed refresh timer count, and a set of counters which are clocked at one half the speed of the processor clock. The control registers are loaded with a value under program control. When the counters reach the value that was programed into the registers, a refresh request is generated. The counters are cleared and begin to increment again. The next arbitration time, a CAS before RAS refresh cycle will be run. There is a safety circuit to detect if a value is written into the refresh timer count which is smaller than the current refresh counter value. If this happens, the counter will be zeroed to prevent a large amount of time between refreshes due to the counters needing to cycle completely around.

DRAM Interface

The DRAM interface consists of RAS, CAS, Write enable, Output enable and multiplexed addresses.

Processor/BusSwitch Interface

The Processor/BusSwitch interface conforms to the MC88110 bus specifications and includes most control signals and the Address lines. Data lines are not included.

ECDM Interface

The ECDM interface consists of the control signals to latch and output data to both the MC88110 bus and DRAM. It includes the buffered address and control signals from the MC88110 bus which pass through the DCAM.

JTAG Test Interface

JTAG boundary scan test is included on each input and output pin.

Scrubbing

Soft error scrubbing circuits are included. There is a scrub count register within the control register section, which is loaded under software control. When a count is placed into the scrub count register, the scrub counter begins to run. When the count in the scrub counter reaches the value in the scrub count register, scrub cycles are started. A cycle is run to each line of memory. The data is brought into the ECDM, checked for errors, single-bit errors corrected and then the line is written back to DRAM. These cycles continue, at a lower priority than all other cycles, until all DRAM is scrubbed. This scrub sequence will repeat itself when the scrub count register again equals the scrub counters.

Scrubbing starts at the highest address of Bank 0 and continues until all of Bank 0 is scrubbed. If Bank 1 is enabled, scrubbing will continue at the highest address of Bank 1 and quit when it reaches the lowest address of Bank 1.

In addition, memory can be scrubbed 1 time by writing the scrub1time bit in the control register to a one. This will cause all of DRAM to be scrubbed once. No more scrubs will take place as a result of this bit until it is written to a 0 and again written to a 1.

DCAM Auto-Program

The DCAM does not rely entirely on system software to program its internal registers. When the DCAM is reset, it searches the serial I^2C bus for a serial EEPROM. Finding this device, the DCAM programs its internal registers from data that has been pre-programmed into the I^2C bus EEPROM. The EEPROM device is a ST24C04 which has two 256 byte pages. The DCAM auto-program data is contained in the upper page of the serial EEPROM.

If the serial EEPROM is not installed on the module, the DCAM programs itself from a set of general default values contained within the DCAM. These default values allow the DCAM to operate over a wide variety of conditions. However, for best performance and to ensure that the DCAM is programmed properly for a particular build type, the DCAM should be programmed automatically at reset from the serial EEPROM installed on the module.

DCAM Software Programming

The DCAM has internal registers that set up basic operating parameters such as refresh that should not be modified by system software. However, many of the DCAM registers are intended to be modified by system software such as enabling desired operating features. When changing the DCAM registers, caution should be taken. It is possible to program the DCAM registers with values that will cause the memory system to not operate or operate improperly. Generally, register bits that affect the timing of the system should never be modified. Table 5-2 describes the DCAM register bits that may be modified by software. Should a value be programmed in the DCAM that causes the system to not function properly, then the DCAM must be reset so that it may auto-program itself from the serial EEPROM.

Since the DCAM does not have a data bus, its internal registers are modified by transmitting data serially via the two wire I^2C bus interface. As described in the *DCAM Auto-Program* section of this chapter, the DCAM programs itself from a serial EEPROM that also resides on the I^2C bus interface. While programming itself, the DCAM is a master on the I^2C bus. Once this operation is complete, the DCAM becomes a slave on the I^2C bus enabling it to respond to programming sequences initiated by the system software.

Once the DCAM has become an I^2C bus slave, it is programmed by software through the I^2C bus master interface in the ECDM. The ECDM I^2C bus master interfaces eases serial programming of the DCAM by handling the I^2C bus protocols in hardware. To software, the programming interface is the ECDM I^2C bus registers. The *ECDM/DCAM I^2C bus Operation* section in the *ECDM* chapter describes the serial bus in more detail and provides programming examples for both the DCAM and the serial EEPROM. The I^{2}C bus address for the DCAM is hC0 and the I^{2}C bus address for the serial EEPROM is hA0. Note that the upper page of the serial EEPROM that contains the DCAM reset defaults is write protected and cannot be modified. However, it is possible to write to the lower page of the serial EEPROM. The *ECDM Control and Status* section in the *ECDM* chapter describes the ECDM I^{2}C bus master interface resisters. Additional I^{2}C bus programming information is contained in the *MVME197BUG 197Bug Debugging Package User's Manual.*

DCAM Programmable Registers

The DCAM register map is shown in the following table.

Table 5-1. DCAM Register Map

Offset	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	1	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	1
2	SL31	SL30	SL29	SL28	SL27	SL26	SL25	DISRAM
3	SH31	SH30	SH29	SH28	SH27	SH26	SH25	SCRUB TIME
4	CAS CLKSL	CAS CLK2	CAS CLK1	PG MODE	ONE BANK	DRAM SIZ3 (16 MEG X 4)	DRAM SIZ2 (4 MEG X 4)	DRAM SIZ1 (1 MEG X 4)
5	REF7	REF6	REF5	REF4	REF3	REF2	REF1	REF0
6	REF TAIL4	REF TAIL3	REF TAIL2	REF TAIL1	REF11	REF10	REF9	REF8
7	KILL CACHE	NOT USED	RD TAIL5	RD TAIL4	RD TAIL3	RD TAIL2	RD TAIL1	RT CLKSL
8	READ ACK7	READ ACK6	READ ACK5	READ ACK4	READ ACK3	READ ACK2	READ ACK1	INTR RUPT
9	SPLIT RAS	READ OE6	READ OE5	READ OE4	READ OE3	READ OE2	READ OE1	NOT USED
10	FEC CKSL	BREAD OE6	BREAD OE5	BREAD OE4	BREAD OE3	BREAD OE2	BREAD OE1	PCG CLKSL
11	PCHG7	PCHG6	PCHG5	PCHG4	PCHG3	PCHG2	PCHG1	PCHG0
12	SLE CDM5	SLE CDM4	SLE CDM3	SLE CDM2	FLE CDM4	FLE CDM3	FLE CDM2	FLE CDM1

Table 5-1. DCAM Register Map (Continued)

Offset	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
13	NOT USED	ERAM OE6	ERAM OE5	ERAM OE4	ERAM OE3	ERAM OE2	ERAM OE1	ROE CLKSL
14	NOT USED	RMWRM OE6	RMWRM OE5	RMWRM OE4	RMWRM OE3	RMWRM OE2	RMWRM OE1	RMW OE5
15	CSR TAIL7	CSR TAIL6	CSR TAIL5	CSR TAIL4	CSR TAIL3	CSR TAIL2	CSR TAIL1	NOT USED
16	BWRT TL4	BWRT TL3	BWRT TL2	BWRT TL1	RMW OE4	RMW OE3	RMW OE2	RMW OE1
17	SEC CLKSL	RMWO CKSL	BWRITE 5	BWRITE 4	BWRITE 3	BWRITE 2	BWRITE 1	WRCLK SEL
18	NOT USED	NOT USED	RMW5	RMW4	RMW3	RMW2	RMW1	NOT USED
19	RMW TAIL7	RMW TAIL6	RMW TAIL5	RMW TAIL4	RMW TAIL3	RMW TAIL2	RMW TAIL71	RMW TLCSL
20	CBRD OE3	CBRD OE2	CBRD OE1	NOT USED	CREAD OE3	CREAD OE2	CREAD OE1	BWR TCSL
21	SC9	SC8	SC7	SC6	SC5	SC4	SC3	SC2
22	SC17	SC16	SC15	SC14	SC13	SC12	SC11	SC10
23	SC25	SC24	SC23	SC22	SC21	SC20	SC19	SC18
24	SC33	SC32	SC31	SC30	SC29	SC28	SC27	SC26
25	NOT USED	NOT USED	NOT USED	CB TAIL4	CB TAIL3	CB TAIL2	CB TAIL1	CBTL CKSL
26	CSR7	CSR6	CSR5	CSR4	NOT USED	NOT USED	NOT USED	NOT USED
27	CSR15	CSR14	CSR13	CSR12	CSR11	CSR10	CSR9	CSR8
28	CSR23	CSR22	CSR21	CSR20	CSR19	CSR18	CSR17	CSR16
29	CSR31	CSR30	CSR29	CSR28	CSR27	CSR26	CSR25	CSR24
30	NOT USED	NOT USED	BRD TAIL5	BRD TAIL4	BRD TAIL3	BRD TAIL2	BRD TAIL1	NOT USED
31								

DCAM Register Set

The DCAM registers are described in the following tables.

Table 5-2. DCAM Register Set: User Registers

Register	Offset	Function	Bits Used	Signals	Valid Values (in Hex)
ID Register	0	Chip ID	8	Read Only	$81
Version Register	1	Chip Version	8	Read Only	1 to current revision
RSAR1	2	RAM Start Address Register, Bank 1	7	SL25-31	Any Valid Address
RSAR2	3	RAM Start Address Register, Bank 2	7	SH25-31	Any Valid Address
CSR Address	26-29	CSR Address Location Register	28	CSR4-31	Any Valid Address
Scrub Count Register	21-24	Scrub Timer Count	32	SC2-33	Any Count
Single Scrub Bit	3	Causes 1 scrub of entire DRAM	1	SCRUB1 TIME	0,1
Disable RAM	2	Disables DRAM and CSR decoding	1	DISRAM	0,1
Interrupt/Ta	8	Selects interrupt or TEA for errors	1	INTRRUPT	0,1
Refresh Count Register	5-6	Refresh Timer Count	12	REF0-11	Any Count

Table 5-3. DCAM Register Set: DRAM Size and Type Registers

Register	Offset	Function	Bits Used	Signals	Valid Values (in Hex)
DRAM Size Register	4	Selects 1 meg by 4, 4 meg by 4, or 16 meg by 4 DRAMs	3	DRAM SIZE 1-3	1,2,4
One Bank	4	Selects one or two banks	1	ONEBANK	0,1
Page Mode	4	Selects Static Column or Page mode DRAMs	1	PGMODE	0,1

Table 5-4. DCAM Register Set: Hardware Setup Registers

Register	Offset	Function	Bits Used	Signals	Valid Values (in Hex)
CAS Clock	4	Selects RAS to CAS time	2	CASCLK1-2	1,2
CAS Clock Select	4	Selects edge of clock for RAS to CAS timing	1	CAS CLKSL	0,1
Precharge	11	Selects RAS high time at end of DRAM cycle	8	PCHG0-7	1 - 7F
Precharge Clock Select	10	Selects edge of clock for RAS high time at end of DRAM cycle	1	PCGCLK SL	0,1
Burst Read Output Enable	10	Output enable ending time for Burst Reads, ECDM to processor	6	BREAD OE1-6	1,2,4,8,10, 20
Kill Cache	7	Stops decode of any cache hits	1	KILL CACHE	0,1
Read Tail	7	Selects RAS low time for normal reads	5	RDTAIL1-5	1,2,4,8,10
Read Tail Clock Select	7	Selects edge of clock used for RAS low time during normal reads	1	RTCLKSL	0,1
Read Ack	8	Selects time from CAS to TA on normal or Burst Read	7	READ ACK1-7	1,2,4,8,10, 20,40
Read Output Enable	9	Output enable ending time for Normal Reads, ECDM to processor	6	READ OE1-6	1,2,4,8,10, 20
Split RAS	9	Splits RAS for 2 Banks	1	SPLIT RAS	0,1
Burst Read Tail	30	Selects RAS low time for burst read cycles	5	BRD TAIL1-5	1,2,4,8,10
Latch First Word in ECDM	12	First word latch timing, all reads	4	FLE CDM1-4	1,2,4,8
Latch First Word in ECDM Clock Select	10	Selects clock edge for first latch timing	1	FECCLKSL	0,1
Latch Second Word in ECDM	12	Second word latch timing, second read (Look Ahead)	4	SLE CDM2-5	1,2,4,8

Table 5-4. DCAM Register Set: Hardware Setup Registers (Continued)

Register	Offset	Function	Bits Used	Signals	Valid Values (in Hex)
Latch Second Word in ECDM Clock Select	17	Selects clock edge for second latch timing	1	SECCLK SL	0,1
End DRAM Output Enable	13	Stops DRAM OE on normal or burst reads	6	ERAMOE OE1-6	1,2,4,8,10, 20
DRAM Output Enable Clock Select	13	Selects edge of clock for DRAM OE on normal or burst reads	1	ROECLK SL	0,1
RMW DRAM Output Enable	14	Stops DRAM OE on read portion of RMW, scrub and cache burst reads	6	RMWRM OE1-6	1,2,4,8,10, 20
RMW ECDM Output Enable	14,16	Selects time to turn on OE from ECDM to DRAM on RMW cycles	5	RMWO1-5	1,2,4,8,10
RMW Output Enable Clock Select	17	Selects edge of clock used to select time to turn on OE from ECDM to DRAM on RMW cycles	1	RMWOE-CKSL	0,1
Burst Write Time Select	17	Selects time for driving the write line to DRAM on burst write cycles	5	BWRITE 1-5	1,2,4,8,10
Burst Write Time Clock Select	17	Selects edge of clock used for driving write line to DRAMs	1	WR CLKSEL	0,1
RMW Tail	19	Selects length of low time of RAS on RMW and scrub cycles	7	RMW TAIL1-7	1,2,4,8,10, 20,40
RMW Tail Clock Select	19	Selects edge of clock used to select length of RAS on RMW and scrub cycles	1	RMWTL CSL	0,1
Refresh Tail	6	Selects length of RAS on refresh cycles	4	REF TAIL1-4	1,2,4,8

Table 5-4. DCAM Register Set: Hardware Setup Registers (Continued)

Register	Offset	Function	Bits Used	Signals	Valid Values (in Hex)
Burst Write Tail	16	Selects length of RAS on burst write cycles	4	BWRT TL1-4	1,2,4,8
Burst Write Tail Clock Select	20	Selects edge of clock used to select length of RAS on burst write cycles	1	BWRTC SL	0,1
Cache Read Output Enable	20	Selects output enable time from ECDM to MC88110 bus on normal cache reads	3	CREAD OE1-3	1,2,4
Cache Burst Read Output Enable	20	Selects output enable time from ECDM to MC88110 bus on burst cache reads	3	CBRD OE1-3	1,2,4
Cache Burst Read Tail	25	Selects length of low time of RAS on cache burst read cycles	4	CB TAIL1-4	1,2,4,8
Cache Burst Read Tail Clock Select	25	Selects edge of clock used to select length of RAS on cache burst read cycles	1	CBTLCKSL	0,1
CSR Tail	15	Selects length of CSR cycles	7	CSR TAIL1-7	1,2,4,8,10, 20,40
RMW Write Line	18	Selects time for write to DRAM on Read-Modify-Write	5	RMW1-5	1,2,4,8,10
Burst Read Tail Select	30	Selects length of RAS on Burst Reads	5	BRD TAIL1-5	1,2,4,8,10

User Operating Registers

The following registers can be used in the normal operation of the DCAM. They are intended to be user configured and used in the normal operation of the DCAM.

ID Register

REG	ID Register							
ADR/SIZ	$FFF01000 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	1	0	0	0	0	0	0	1
OPER	R	R	R	R	R	R	R	R
RESET	$81	$81	$81	$81	$81	$81	$81	$81

Offset = 0. This is a read-only register which has the code to identify the type chip. The DCAM code is $81.

Version Register

REG	Version Register							
ADR/SIZ	$FFF01001 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	0	0	0	0	0	0	0	1
OPER	R	R	R	R	R	R	R	R
RESET	01	01	01	01	01	01	01	01

Offset = 1. This is a read-only register which has the code to identify the version of the chip. This will start with the value of 01 and be incremented every time the chip is respun with any change.

RAM Start Address Register 1 - RSAR1 (SL25-31)

REG	RAM Start Address Register, Bank 1							
ADR/SIZ	$FFF01002 (7 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	SL31	SL30	SL29	SL28	SL27	SL26	SL25	
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
RESET	0	0	0	0	0	0	0	

Offset = 2. The RAM Start Address Register 1 holds the beginning address of RAM for bank 1. For a single bank application, this will be the only address used. The RAM size is decoded in the DRAM Size register, which will determine the ending address of RAM. If the DCAM is programed for 1 meg by four DRAMs, all seven bits will be decoded. If the DCAM is programed for 4 meg by four DRAMs, SL25 and SL26 will be ignored. If the DCAM is programed for 16 meg by four DRAMs, SL25, SL26, SL27, and SL28 will be ignored.

Disable RAM Bit - DISRAM

REG	Disable RAM Bit							
ADR/SIZ	$FFF01002 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD								DISRAM
OPER								R/W
RESET								0

Offset = 2. This bit, when a 0 (default) causes DRAM not to be decoded. When a 1, decoding is allowed.

RAM Start Address Register 2 - RSAR2 (SH25-31)

REG	RAM Start Address Register, Bank 2							
ADR/SIZ	$FFF01003 (7 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	SH31	SH30	SH29	SH28	SH27	SH26	SH25	
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
RESET	0	0	0	0	0	0	0	

Offset = 3. The RAM Start Address Register 2 holds the beginning address of RAM for bank 2. The RAM size is decoded in the DRAM Size register, which will determine the ending address of RAM. When this register has the same value as RAM Start Address Register 1, this register will never decode. If the DCAM is programed for 1 meg by four DRAMs, all seven bits will be decoded. If the DCAM is programed for 4 meg by four DRAMs, SH25 and SH26 will be ignored. If the DCAM is programed for 16 meg by four DRAMs, SH25, SH26, SH27, and SH28 will be ignored.

Single Scrub Bit - SCRUB1TIME

REG	Single Scrub Bit							
ADR/SIZ	$FFF01003 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD								SCRUB1 TIME
OPER								R/W
RESET								0

Offset = 3. The Single Scrub Bit is used to cause a scrub of the entire DRAM to take place. When set to a 1, a scrub request will be issued. A scrub of DRAM will begin. No further action will take place until the single scrub bit is returned to a 0 and again brought to a 1.

Refresh Count Register - (REF0-11)

REG	Refresh Count Register							
ADR/SIZ	$FFF01005 (8 bits)							
BIT	7	6	5	4				0
FIELD	REF7	REF6	REF5	REF4	REF3	REF2	REF1	REF0
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	$69	$69	$69	$69	$69	$69	$69	$69

REG	Refresh Count Register							
ADR/SIZ	$FFF01006 (8 bits)							
BIT	7	6	5	4				0
FIELD					REF11	REF10	REF9	REF8
OPER					R/W	R/W	R/W	R/W
RESET					$69	$69	$69	$69

Offset = 5 and 6. The Refresh Count Register holds a count which determines how often a refresh cycle will take place. The Refresh Register will increment at a rate of one half of the frequency applied to the clock pin. When the value in the Refresh Register and the Refresh Count Register are equal, a refresh request will be issued and the Refresh Register will be cleared. The Refresh Register will again begin to count and when the arbiter allows a refresh cycle, the refresh cycle will be preformed. The following chart illustrates sample refresh times.

Count in Refresh Count Register

FREQ	$40	$50	$60	$69	$70	$80	$90	$A0	$B0	$C0
50 MHz	5.1	6.4	7.6	8.4	8.9	10.2	11.5	12.8	14	15.3
45 MHz	5.6	7.1	8.5	9.3	9.9	11.3	12.8	14.2	15.6	17
40 MHz	6.4	8	9.6	10.5	11.2	12.8	14	16	17.6	
35 MHz	7.3	9.1	11	11.9	12.7	14.6	16.4	18.2		
33 MHz	7.7	9.7	11.7	12.7	13.5	15.5	17.4			

To find other values not shown in the chart, use the following formula:

1 divided by freq) times 2) times the decimal value for the hex number in the Refresh Count Register. This is the value in seconds. Then multiply by 1,000,000 for time in microseconds.

Interrupt/TA Bit - INTRRUPT

REG	Interrupt/TA Bit							
ADR/SIZ	$FFF01008 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD								INTER RUPT
OPER								R/W
RESET								0

Offset = 8. This bit, when a 1 (default) will give an interrupt and cause the TA signal to be when there is a double-bit error. When this bit is a 0 it will cause the TEA signal to be driven when there is a double-bit error. Read cycles will be faster when using the interrupt mode. When changing to the TEA mode, the cycle times must be extended to allow for propagation of the error signals through the ECDM and DCAM. This requires a later dtack, and depending on the DRAMs and clock speed, could require other changes.

Scrub Count Register - (SC2-32)

REG	Scrub Count Register							
ADR/SIZ	$FFF01021 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	SC9	SC8	SC7	SC6	SC5	SC4	SC3	SC2
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0

REG	Scrub Count Register							
ADR/SIZ	$FFF01022 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	SC17	SC16	SC15	SC14	SC13	SC12	SC11	SC10
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0

REG	Scrub Count Register							
ADR/SIZ	$FFF01023 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	SC25	SC24	SC23	SC22	SC21	SC20	SC19	SC18
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0

REG	Scrub Count Register							
ADR/SIZ	$FFF01024 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	SC33	SC32	SC31	SC30	SC29	SC28	SC27	SC26
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0

Offset = 21 through 24. The Scrub Count Register holds a count which determines if and when a scrub of RAM will take place. When the register is written to a value of 0, no scrubs will take place. Any value of more than 0 will cause the scrub counter to begin to count. It will count at one half the clock frequency applied to the clock pin. When that value reaches the value placed in the Scrub Count Register, a scrub of all of RAM will begin. The scrub counter will reset to 0 and again begin to count. This process will continue as long as a value of more than 0 is in the Scrub Count Register. By calculating the proper values to place in the Scrub Count Register, scrubs can be preformed at specified intervals such as once an hour or once a day.

CSR Address Mapping Register - (CSR 4-31)

REG	CSR Address Mapping Register							
ADR/SIZ	$FFF01026 (4 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	CSR7	CSR6	CSR5	CSR4				
OPER	R/W	R/W	R/W	R/W				
RESET	0	0	0	0				

REG	CSR Address Mapping Register							
ADR/SIZ	$FFF01027 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	CSR15	CSR14	CSR13	CSR12	CSR11	CSR10	CSR9	CSR8
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0

REG	CSR Address Mapping Register							
ADR/SIZ	$FFF01028 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	CSR23	CSR22	CSR21	CSR20	CSR19	CSR18	CSR17	CSR16
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0

REG	CSR Address Mapping Register							
ADR/SIZ	$FFF01029 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	CSR31	CSR30	CSR29	CSR28	CSR27	CSR26	CSR25	CSR24
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0

Offset = 26 through 29. The CSR Address Mapping Register maps the location of the CSR on the processor bus. There is no restriction as to where it can be mapped. When a new address is written to this register it will change where the CSR is located.

5

DRAM Size and Type Setup Registers

The DRAM size and setup registers are intended to configure the system for the type and amount of DRAM present at power on and not be changed by software. In normal operation software can make use of this information to assist in things like memory sizing.

Page Mode Bit - PGMODE

REG	Page Mode Bit							
ADR/SIZ	$FFF01004 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD				PG MODE				
OPER				R/W				
RESET				1				

Offset = 4. The Page Mode Bit selects either page mode DRAMs with value 1, (Default) or static column DRAMs with the value of 0. The system will operate with page mode set and static column DRAMs installed. It will not function with static column mode set and page mode DRAMs installed.

One Bank Bit

REG	One Bank Bit							
ADR/SIZ	$FFF01004 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD					ONE BANK			
OPER					R/W			
RESET					1			

Offset = 4. This bit, when a 1 selects decoding for one bank (default) or, when a 0, selects decoding for two banks.

DRAM Size Register

REG	DRAM Size Register							
ADR/SIZ	$FFF01004 (3 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD						DSIZ3	DSIZ2	DSIZ1
OPER						R/W	R/W	R/W
RESET						1	1	1

Offset = 4. This register holds the size of DRAMs installed. Only one bit may be set. Bit 0 set = 1 MB by 4 DRAMs. Bit 1 set = 4 MB by 4 DRAMs. Bit 2 set = 16 MB by 4 DRAMs installed. The system will function with the bits set to indicate a smaller size DRAM installed than is actually in the system. The additional DRAM will not be seen.

The following chart will give the total RAM size based on the selections programed in:

	1 MB by 4	4 MB by 4	16 MB by 4
1 bank	32 megabytes	128 megabytes	512 megabytes
2 banks	64 megabytes	256 megabytes	1024 megabytes

Hardware Setup Registers

The Hardware Setup Registers are the registers to configure the general chip functions to the clock speed, DRAM type, and DRAM speed for a particular hardware combination. They are intended to be set up to an initial configuration on powerup reset and not changed by software again. This is an engineering function in the same sense as the logic design and requires engineering support to calculate and verify the values to be inserted.

CAS Clock Select Bit - CASCLKSL

REG	CAS Clock Select Bit							
ADR/SIZ	$FFF01004 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD	CAS CLKSL							
OPER	R/W							
RESET	0							

Offset = 4. The CAS Clock Select Bit will select the rising or falling edge of the clock to give a finer adjustment of the time between the start of RAS and the start of CAS. RAS will start as a function of a rising edge, and a 0 in CASCLKSL will start the time from RAS to CAS on the next falling edge of the clock and a 1 in CASCLKSL will start the time from RAS to CAS on the next rising edge of the clock.

CAS Clock Bits - (CASCLK1-2)

REG	CAS Clock Bits							
ADR/SIZ	$FFF01004 (2 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD		CAS CLK2	CAS CLK1					
OPER		R/W	R/W					
RESET		0	0					

Offset = 4. The CAS Clock Bits will select the time between RAS going low to the time CAS goes low.

Refresh Tail Register - (REFTAIL1-4)

REG	Refresh Tail Register							
ADR/SIZ	$FFF01006 (4 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	REF TAIL4	REF TAIL3	REF TAIL2	REF TAIL1				
OPER	R/W	R/W	R/W	R/W				
RESET	4	4	4	4				

Offset = 6. The Refresh Tail Register selects the number of clocks from the time RAS goes low to the time RAS goes high in a refresh cycle.

Kill Cache Bit

REG	Kill Cache Bit							
ADR/SIZ	$FFF01007 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD	KILL CACHE							
OPER	R/W							
RESET	0							

Offset = 7. Kill Cache disables any cache hit from being decoded and forces all reads to be a DRAM access.

Read Tail Select Register - (RDTAIL1-5)

REG	Read Tail Select Register							
ADR/SIZ	$FFF01007 (5 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD			RD TAIL5	RD TAIL4	RD TAIL3	RD TAIL2	RD TAIL1	
OPER			R/W	R/W	R/W	R/W	R/W	
RESET			02	02	02	02	02	

Offset = 7. The Read Tail Select Register selects the number of clocks that RAS is low after CAS goes low on a normal read cycle.

Read Tail Clock Select Bit - RTCLKSL

REG	Read Tail Clock Select Bit							
ADR/SIZ	$FFF01007 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD								RT CLKSL
OPER								R/W
RESET								1

Offset = 7. The Read Tail Clock Select Bit selects which edge of the clock to use when computing the read tail time on a normal read cycle This gives one half clock resolution for the length of RAS low time. A 0 in RTCLKSL will start the time to end RAS on the next falling edge of the clock and a 1 in RTLCLKSL will start the time to end RAS on the next rising edge of the clock.

Read Acknowledge Select Register - (READACK1-7)

REG	Read Acknowledge Select Register							
ADR/SIZ	$FFF01008 (7 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	READ ACK7	READ ACK6	READ ACK5	READ ACK4	READ ACK3	READ ACK2	READ ACK1	
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
RESET	02	02	02	02	02	02	02	

Offset = 8. The Read Acknowledge Select Register selects the time from CAS going low to the time Transfer Acknowledge is sent to the processor on a read or burst read.

Split RAS Bit

REG	Split RAS Bit							
ADR/SIZ	$FFF01009 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD	SPLIT RAS							
OPER	R/W							
RESET	0							

Offset = 9. Split RAS selects whether all RAS lines are driven at the same time (0), or if only RAS0-RAS7 are driven when Bank 0 is decoded and only RAS8-RAS15 are driven when Bank 1 is decoded.

Read Output Enable Select Register - (READOE1-6)

REG	Read Output Enable Select Register							
ADR/SIZ	$FFF01009 (6 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD		READ OE6	READ OE5	READ OE4	READ OE3	READ OE2	READ OE1	
OPER		R/W	R/W	R/W	R/W	R/W	R/W	
RESET		20	20	20	20	20	20	

Offset = 9. The Read Output Enable Select Register selects the length of time until the output enable is turned off from the ECDM to the processor bus on a normal read.

Latch First Word in ECDM Clock Select Bit - FECCLKSL

REG	Latch First Word In ECDM Clock Select Bit							
ADR/SIZ	$FFF01010 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD	FEC CKSL							
OPER	R/W							
RESET	1							

Offset = 10. The FECCLKSL bit selects either the rising or falling edge of the clock to count the FLECDM time to give more resolution. A 0 selects the rising edge, and 1 selects the falling edge.

Burst Read Output Enable End Time Select Register - (BREADOE1-6)

REG	Burst Read Output Enable End Time Select Register							
ADR/SIZ	$FFF01010 (6 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD		BREAD OE6	BREAD OE5	BREAD OE4	BREAD OE3	BREAD OE2	BREAD OE1	
OPER		R/W	R/W	R/W	R/W	R/W	R/W	
RESET		10	10	10	10	10	10	

Offset = 10. The Burst Read Output Enable End Time Select Register selects how many clocks are used before turning off the drivers from the ECDM to the processor bus.

Precharge Clock Select Bit - PCGCLKSL

REG	Precharge Clock Select Bit							
ADR/SIZ	$FFF01010 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD								PCG CLKSL
OPER								R/W
RESET								1

Offset = 10. The Precharge Clock Select Bit will select the rising or falling edge of the clock to count the time for the precharge, giving one half clock increments to fine tune the precharge time for high speed operations. A 1 will select the rising edge of the clock and a 0 will select the falling edge of the clock.

Sample Precharge times - time in chart is precharge time in nanoseconds.

Count in Precharge Register

FREQ	$1	$10	$20	$40	$50	$80	$90	$A0	$C0	$FF
50 MHz	40	340	660	1300	1620	2580	2900	3220	3860	5120
45 MHz	44	377	733	1444	1800	2866	3221	3577	4288	5688
40 MHz	50	425	825	1625	2025	3225	3625	4025	4825	6400
35 MHz	57	485	940	1852	2308	3676	4132	4588	5500	7296
33 MHz	60	570	990	1950	2430	3870	4350	4830	5790	7680

Precharge Register - (PCHG0-7)

REG	Precharge Register							
ADR/SIZ	$FFF01011 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	PCHG7	PCHG6	PCHG5	PCHG4	PCHG3	PCHG2	PCHG1	PCHG0
OPER	R	R	R	R	R	R	R	R
RESET	01	01	01	01	01	01	01	01

Offset = 11. The Precharge Register bits are 8 bits which are put into a register and compared to the value in the precharge counter. The precharge time is the time RAS must remain high at the end of a DRAM cycle. When the results of the 8-bit precharge counter match the PRECHARGE value, the cycle is ended and a new cycle can begin. This value is normally a very low value and is not changed by operating software in most operations. It's prime use, other than setting the minimum time RAS must be high for a certain clock speed/DRAM combination, is to slow the system operation when power is a factor. This might be for a lap top or other battery operation.

Refer to the precharge time chart in the *Precharge Clock Select* section for times for various clock speeds and PRECHARGE values.

Latch Second Word in ECDM Register - (SLECDM2-5)

REG	Latch Second Word In ECDM Register							
ADR/SIZ	$FFF01012 (4 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	SL ECDM5	SL ECDM4	SL ECDM3	SL ECDM2				
OPER	R/W	R/W	R/W	R/W				
RESET	0	0	0	0				

Offset = 12. The Latch Second Word In ECDM Register selects the number of clocks from RAS going low until the second line is latched into the ECDM. At the same time the data is latched, the CAS addresses are incremented to the next line.

Latch First Word in ECDM Register - (FLECDM1-4)

REG	Latch First Word In ECDM Register							
ADR/SIZ	$FFF01012 (4 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD					FL ECDM4	FL ECDM3	FL ECDM2	FL ECDM1
OPER					R/W	R/W	R/W	R/W
RESET					4	4	4	4

Offset = 12. The Latch First Word In ECDM Register selects the number of clocks from RAS going low until the first line is latched into the ECDM. At the same time the data is latched, the CAS addresses are incremented to the next line.

End RAM Output Enable Register - (ERAMOE1-6)

REG	End RAM Output Enable Register							
ADR/SIZ	$FFF01013 (6 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD		ERAM OE6	ERAM OE5	ERAM OE4	ERAM OE3	ERAM OE2	ERAM OE1	
OPER		R/W	R/W	R/W	R/W	R/W	R/W	
RESET		20	20	20	20	20	20	

Offset = 13. The End RAM Output Enable Register selects the time from latching of the cycle type to the time DRAM output enable is turned off. This is used in normal and burst reads.

RAM Output Enable Clock Select Bit - ROECLKSL

REG	RAM Output Enable Clock Select Bit							
ADR/SIZ	$FFF01013 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD								ROE CLKSL
OPER								R/W
RESET								1

Offset = 13. The RAM Output Enable Clock Select Bit is used to select the rising or falling edge of the clock to count the clock cycles to turn off the output enable of DRAM on normal or burst reads, or read-modify-write cycles.

Read-Modify-Write (normal write) RAM Output Enable Register- (RMWRMOE1-6)

REG	Read-Modify-Write (normal write) RAM Output Enable Register							
ADR/SIZ	$FFF01014 (6 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD		RMWRM OE6	RMWRM OE5	RMWRM OE4	RMWRM OE3	RMWRM OE2	RMWRM OE1	
OPER		R/W	R/W	R/W	R/W	R/W	R/W	
RESET		20	20	20	20	20	20	

Offset = 14. The Read-Modify-Write RAM Output Enable Register selects the time from latching of the cycle type to the time DRAM output enable is turned off. This is used in normal (non burst) writes, scrub, and cache burst reads.

CSR Tail Select Register - (CSRTAIL1-7)

REG	CSR Tail Select Register (CSRTAIL1-7)							
ADR/SIZ	$FFF01015 (7 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	CSR TAIL7	CSR TAIL6	CSR TAIL5	CSR TAIL4	CSR TAIL3	CSR TAIL2	CSR TAIL1	
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
RESET	02	02	02	02	02	02	02	

Offset = 15. The CSR Tail Select Register selects the time from the latching of the CSR cycle until the time the precharge phase is started. This allows a broad selection of time for the ECDM to get data on to the processor bus before TA is sent to the processor.

Read-Modify-Write (normal write) ECDM Output Enable Register - (RMWOE1-5)

REG	Read-Modify-Write (normal write) ECDM Output Enable Register							
ADR/SIZ	$FFF01014 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD								RMW OE5
OPER								R/W
RESET								4

REG	Read-Modify-Write (normal write) ECDM Output Enable Register							
ADR/SIZ	$FFF01016 (4 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD					RMW OE4	RMW OE3	RMW OE2	RMW OE1
OPER					R/W	R/W	R/W	R/W
RESET					4	4	4	4

Offset = 14 and 16. The Read-Modify-Write ECDM Output Enable Register selects the number of clocks from latching of the cycle type to the time ECDM to DRAM output enable is turned on. This is used in normal (non burst) writes and scrub.

Burst Write Tail Register - (BWRTTL1-4)

REG	Burst Write Tail Register							
ADR/SIZ	$FFF01016 (4 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	BWRT TL4	BWRT TL3	BWRT TL2	BWRT TL1				
OPER	R/W	R/W	R/W	R/W				
RESET	2	2	2	2				

Offset = 16. The Burst Write Tail Register selects the number of clocks from the time CAS goes low to the time RAS goes high in a burst write cycle.

Latch Second Word in ECDM Clock Select Bit - SECCLKSL

REG	Latch Second Word In EDCM Clock Select Bit							
ADR/SIZ	$FFF01017 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD	SEC CKSL							
OPER	R/W							
RESET	1							

Offset = 17. The SECCLKSL bit selects either the rising or falling edge of the clock to count the SLECDM time to give more resolution. A 1 selects the rising edge, and 0 selects the falling edge.

Read-Modify-Write (normal write) ECDM Output Enable Clock Select Bit - RMWOCKSL

REG	Read-Modify-Write (normal write) EDCM Output Enable Clock Select Bit							
ADR/SIZ	$FFF01017 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD		RMWO CKSL						
OPER		R/W						
RESET		0						

Offset – 17. The Read-Modify-Write ECDM Output Enable Clock Select Bit is used to select either the rising or falling clock edge to count to the time OEECDM goes true.

5

Burst Write Time Select Register - (BWRITE1-5)

REG	Burst Write Time Select Register							
ADR/SIZ	$FFF01017 (5 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD			BWRITE5	BWRITE4	BWRITE3	BWRITE2	BWRITE1	
OPER			R/W	R/W	R/W	R/W	R/W	
RESET			04	04	04	04	04	

Offset = 17. The Burst Write Time Select Register selects the number of clocks from CAS going low to the time the write line to the DRAMs goes low in a burst write cycle.

Burst Write Time Clock Select Bit - WRCLKSEL

REG	Burst Write Time Clock Select Bit							
ADR/SIZ	$FFF01017 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD								WR CLKSEL
OPER								R/W
RESET								1

Offset = 17. The Burst Write Time Clock Select Bit is used to select either the rising or falling clock edge to count to the time BWRITE goes true.

Read-Modify-Write Write Time Select Register - (RMW1-5)

REG	Read-Modify-Write Write Time Select Register							
ADR/SIZ	$FFF01018 (5 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD			RMW5	RMW4	RMW3	RMW2	RMW1	
OPER			R/W	R/W	R/W	R/W	R/W	
RESET			04	04	04	04	04	

Offset = 18. The Read-Modify-Write Write Time Select Register selects the time the write line to the DRAMs is lowered on a read-modify-write (normal write) and a scrub cycle.

Read-Modify-Write (normal write) Tail Register - (RMWTAIL1-7)

REG	Read-Modify-Write (normal write) Tail Register							
ADR/SIZ	$FFF01019 (7 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	RMW TAIL7	RMW TAIL6	RMW TAIL5	RMW TAIL4	RMW TAIL3	RMW TAIL2	RMW TAIL1	
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
RESET	10	10	10	10	10	10	10	

Offset = 19. The Read-Modify-Write Tail Register selects the number of clocks from CAS going low to the time RAS goes back high in a read-modify-write (normal write) or scrub cycle.

Read-Modify-Write (normal write) Tail Clock Select Bit - RMWTLCSL

REG	Read-Modify-Write (normal write) Tail Clock Select Bit							
ADR/SIZ	$FFF01019 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD								RMW TLCSL
OPER								R/W
RESET								1

Offset = 19. The Read-Modify-Write Tail Clock Select Bit is used to select either the rising or falling clock edge to count to the time from CAS going low to the time RAS goes back high in a normal write or scrub cycle. A 0 selects the rising edge, and 1 selects the falling edge.

Cache Burst Read Output Enable Select Register-(CBRDOE1-3)

REG	Cache Burst Read Output Enable Select Register							
ADR/SIZ	$FFF01020 (3 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	CBRD OE3	CBRD OE3	CBRD OE3					
OPER	R/W	R/W	R/W					
RESET	1	1	1					

Offset = 20. The Cache Burst Read Output Enable Select Register selects the length of time that the ECDM output enable to the processor bus is turned on during a cache burst read cycle.

Cache Read Output Enable Register - (CREADOE1-3)

REG	Cache Read Output Enable Register							
ADR/SIZ	$FFF01020 (3 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD					CREAD OE3	CREAD OE2	CREAD OE1	
OPER					R/W	R/W	R/W	
RESET					1	1	1	

Offset = 20. The Cache Read Output Enable Register selects the turn off time of the data drivers from the ECDM to the system bus in a normal cache normal read cycle.

Burst Write Tail Clock Select Bit - BWRTCSL

REG	Burst Write Tail Clock Select Bit							
ADR/SIZ	$FFF01020 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD								BWR TCSL
OPER								R/W
RESET								1

Offset = 20. The Burst Write Tail Clock Select Bit is used to select either the rising or falling clock edge to count to the time from CAS going low to the time RAS goes back high in a burst write cycle. A 1 selects the rising edge and 0 selects the falling edge.

Cache Burst Read Tail Register - (CBTAIL1-4)

REG	Cache Burst Read Tail Register							
ADR/SIZ	$FFF01025 (4 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD				CBTAIL4	CBTAIL3	CBTAIL2	CBTAIL1	
OPER				R/W	R/W	R/W	R/W	
RESET				1	1	1	1	

Offset = 25. The Cache Burst Read Tail Register selects the number of clocks from CAS going low to RAS going high in a cache burst read cycle.

Cache Burst Read Tail Clock Select Bit - CBTLCKSL

REG	Cache Burst Read Tail Clock Select Bit							
ADR/SIZ	$FFF01025 (1 bit)							
BIT	7	6	5	4	3	2	1	0
FIELD								CBTL CKSL
OPER								R/W
RESET								0

Offset = 25. The Cache Burst Read Tail Clock Select Bit is used to select either the rising or falling clock edge to count to the time from CAS going low to the time RAS goes back high in a cache burst read cycle.

Burst Read Tail Select Register - (BRDTAIL1-5)

REG	Burst Read Tail Select Register							
ADR/SIZ	$FFF01030 (5 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD			BRD TAIL5	BRD TAIL4	BRD TAIL3	BRD TAIL2	BRD TAIL1	
OPER			R/W	R/W	R/W	R/W	R/W	
RESET			04	04	04	04	04	

Offset = 30. The Burst Read Tail Select Register selects the length of time that RAS stays low after CAS goes low on a burst read cycle.

Functional Controls

Sleep Mode Pin

The sleep mode pin, when high, allows normal operation. When low, all functions in the DCAM which are not necessary to keep data refreshed in the DRAM are shut down. Refresh cycles are performed at the normal interval time as programed into the part, however, no cycles are decoded, no scrub cycles are counted or run. The purpose of this is to conserve power when operating in a battery powered mode. Uses include going to the sleep mode when not in operation, or after a fixed time to conserve power when not being heavily used. This signal is internally synchronized with the clock and will finish the cycle in progress before going into the sleep mode.

External Refresh

The external refresh pin chooses either the internal counters or the external pin to cause refreshes. When this pin is high, no refreshes will be performed. When taken low for at least two and not more than 6 clock periods, one refresh will be performed as soon as the current cycle is finished, or immediately if none is in progress. After being held low for at least 6 clock cycles, refresh control is transferred to internal counters and refreshes are preformed automatically as the time programed into the refresh counters is reached. Once the refresh pin is taken low and the internal refresh mode is used the pin must be kept low to stay in the internal refresh mode. If it is taken high, the part will again change into the external refresh control mode. This signal must be changed more than 5 nanoseconds before or after the rising edge of the clock to ensure synchronous operation.

Disable In/Disable Out

There are two input pins and one output pin associated with the disable function. Inputs "DISABLE" and "ROM0" are ORed together inside the DCAM and either will cause decoding of the DRAM and CSR to be disabled. Either being true (driven to a 0) on the input will cause the output "DISOUT" to be driven low. In addition, "DISOUT" will be driven low until the CSR address in the memory map (offset 26 though 29) has been changed from the default location. This is used for situations where there are multiple DCAMs in a system. When a system is first powered up with more than one DCAM they will all be mapped into the same locations. The ROM0 signal is used to disable all DCAMs until vector fetches are done. At this time the "DISOUT" will still be low, disabling any following DCAMs. All DRAM will be at 0 and the CSRs will be at $FFF01000. Making use of the "DISOUT" pin, one DCAM at a time can be re-mapped without conflict. Memory from the first DCAM is mapped to a new location, then the CSR is moved to a new location, after which, the original CSR location is read again. If there is a response, a second DCAM has been found. The same procedure is repeated until there is no response at the CSR address. At this time, all DCAMs have been located.

ERROR CORRECTION AND DATA MULTIPLEXER (ECDM) 6

Introduction

This chapter describes the Error Correction and Data Multiplexer, an ASIC designed for the MVME197 series of Single Board Computers, and referred to as the ECDM. The ECDM is the second generation of this device type and incorporates many new features.

Overview

The ECDM is designed specifically to support cache line bursts for the MC88110 RISC microprocessor or the MC88410 cache controller. However, the ECDM is also compatible with the MC68040 microprocessor and any other design environment that emulates these bus interfaces. In addition, the ECDM provides an advanced Error Detection And Correction (EDAC) system that provides single-bit correction of data errors independently for each ECDM device. This means that four ECDM devices, such as in a 64-bit implementation, provide four bits of error correction. Memory devices in a by-four configuration may be used and the system will maintain data integrity even when an entire memory device fails. Configurations of 32 bits (2 ECDM devices) do not provide this same protection. However, 2 of the 4 bits in any memory device may fail and the system will still maintain data integrity. Multiple data errors that can not be corrected are detected by the error correction circuitry and signals are output that may be used to notify the processor by a bus error or interrupt.

The ECDM is a 0.8 micron technology CMOS gate array that is housed in a 160-pin Plastic Quad Flat Pack (PQFP). The device is also packaged in a 324-pin LGA package to further reduce the device size.

ECDM Features

These are some of the features of the ECDM gate array:

- ❑ Supports n/1/1/1...., 4 or 8 word cache line bursts
- ❑ Two 64-bit write data latches
- ❑ Two 64-bit read data latches with separate latch enables
- ❑ Monolithic Error Detection And Correction circuitry
- ❑ Single-bit error correction, multiple-bit detection

- ❏ Error logger and error indication output signals
- ❏ By-four memory device error correction in 64-bit configurations (four ECDM devices)
- ❏ On-chip cache line address counter
- ❏ Write back of corrected data for error scrubbing
- ❏ High speed, low power CMOS technology
- ❏ Automatic byte selection encoding
- ❏ Diagnostic Control and Status Register (CSR)
- ❏ I^{2}C bus serial data master interface
- ❏ Boundary Scan test capability

General Description

The main function of the ECDM is to provide the latches and multiplexers that enable fast burst data transfers between the main memory and a processor's internal cache memories. The ECDM performs this function and at the same time generates check-bits, according to a Modified Hamming Code, that are stored with the data. The check-bits along with the data are combined in parity trees that determine the encoding of data error bit locations.

The ECDM latches write data and generates check-bit data on write cycles and latches read data along with the check-bits on read cycles. Furthermore, on read cycles, the ECDM directs the data and check-bit information to a syndrome generator that produces a code that indicates no error, a single-bit error, or a multiple-bit error. The syndrome code is presented to a decoder that produces a signal that inverts a data bit in error or, in the case of non-correctable errors, outputs an error signal that may be used to bus error the processor. In either case, the error may be logged allowing system diagnostic software the ability to notify system services of errors or warn of faulty components.

The ECDM has an I^{2}C bus (Inter-Integrated Circuit Communications bus) two wire serial interface. This interface may be used to communicate configuration information to a slave I^{2}C device such as the DCAM (Dynamic RAM Controller and Address Multiplexer). The DCAM has an address interface but not a data interface. The ECDM has a data interface and a limited address interface. The I^{2}C bus is used to communicate data between the ECDM and DCAM in a pin efficient manner. The DCAM provides the ECDM CSR selection (CSRCS*) and the ECDM provides the data registers necessary to implement the I^{2}C controller. In this configuration, the DCAM is not required to have a data interface and the ECDM does not need a complete address interface. Data is transmitted serially between the devices. Therefore, programmable control of configuration and status for each device is provided without requiring each device to have a large number of pins for this feature.

The I^{2}C bus is a standard interface that has multi-master capability and is supported with a number of "off the shelf" devices such as serial EEPROMS. The ECDM I^{2}C interface is compatible with these devices and the inclusion of a serial EEPROM in the memory CSR subsystem may be desirable. The EEPROM could maintain configuration information related to the memory subsystem even when power is removed from the system. Nothing prevents the use of the EEPROM for storing other system variables that must be maintained with the loss of power.

ECDM Block Diagram

The following diagram represents the ECDM and shows data flow to/from the processor on the left and the memory data busses on the right. Note that the processor bus (D bus) is 16 bits wide and the memory busses (RA, RB, RC, RD busses) total 64 bits (4 x 16). An 8-bit check-bit (CD bus) bus provides the data path to and from the check-bit memory devices.

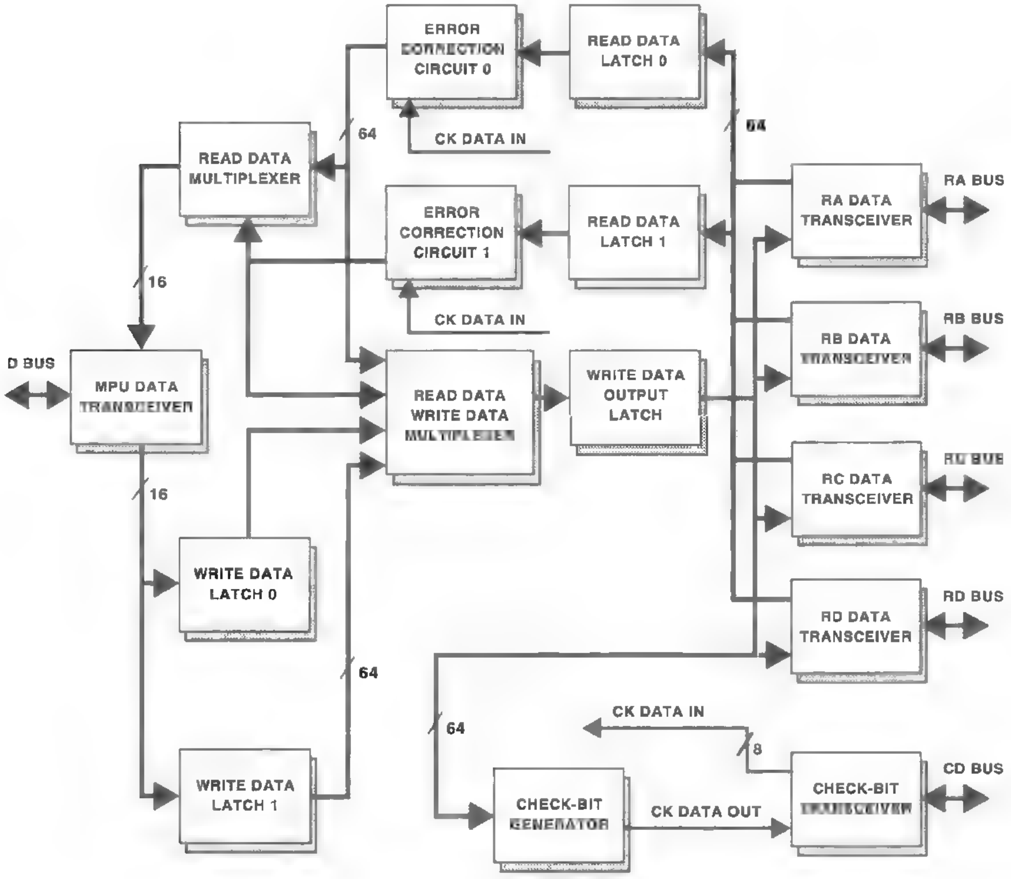


Figure 6-1. ECDM Block Diagram

ECDM Hamming Code

The ECDM utilizes a Modified Hamming Code which determines the encoding of the parity matrix. This is a SEC-DED (single error correct, double error detect) code that is based on a 64-bit data word. Eight check-bits are required to implement single-bit error correction and double-bit detection on a 64-bit data word. Although this is a SEC-DED code, it has a high probability of detecting multiple-bit errors as well. The following figure illustrates the ECDM modified Hamming code.

	RD																RC															
ECDM	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
BYTE	7								6								5								4							
DATA	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
S0				X								X			X	X				X	X	X			X	X	X					
S1	X			X								X								X			X	X				X	X	X		
S2		X			X			X	X			X								X								X			X	X
S3			X			X	X			X			X			X	X			X								X				
S4	X	X	X	X	X	X	X	X			X			X	X			X			X			X	X			X				
S5	X	X	X						X	X	X	X	X	X	X	X			X			X	X			X			X			X
S6				X	X	X			X	X	X						X	X	X	X	X	X	X	X			X			X	X	
S7				X			X	X				X	X	X			X	X	X						X	X	X	X	X	X	X	X

	RB																RA																							
ECDM	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15								
BYTE	3								2								1								0								CHECK-BITS							
DATA	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	C0	C1	C2	C3	C4	C5	C6	C7
S0	X	X	X	X	X	X	X	X			X			X	X			X			X			X	X			X					X							
S1	X	X	X						X	X	X	X	X	X	X	X			X			X	X			X			X			X		X						
S2				X	X	X			X	X	X						X	X	X	X	X	X	X	X		X				X	X	X			X					
S3				X			X	X				X	X	X			X	X	X						X	X	X	X	X	X	X	X				X				
S4				X								X			X	X				X	X	X			X	X	X										X			
S5	X			X								X								X			X	X				X	X	X								X		
S6		X			X			X	X			X								X								X			X	X							X	
S7			X			X	X			X			X			X	X			X								X												X

Figure 6-2. ECDM Modified Hamming Code

The code is divided into four groups (A through D) of 16 data bits. On writes, the ECDM presents the write data to the parity tree in each syndrome row (S7-S0) as indicated by an X under each data bit column. The output of these 8 parity trees is the check-bits (C7-C0). On reads, the ECDM latches the data corresponding to each group (A through D) and combines it to form a 64-bit data word. This 64-bit data word, along with the 8 check-bits, are presented to the parity tree in each syndrome row as indicated by an X under each data bit column. Bit failures produce different parity then is expected in each parity tree and it is the difference of this parity information that enables the EDAC system to decode bit errors from the syndrome code.

Each ECDM is an independent EDAC system. When multiple ECDMs are used in a system, there are no interconnect requirements from one device to another. This method provides for fast error checking since the devices are not required to wait for input from each other.

ECDM Error Logging

The ECDM has error logger status registers to more easily identify memory faults. The ECDM's always log errors in their status registers. However, they may also be programmed to lock the error status depending on error type (correctable or non-correctable) or assert error output signals such as NCER* (Non-Correctable ERror) or ERLOG* (ERror Log).

NCER* is connected to the DCAM and is used for two functions. 1) To prevent writes to memory when non correctable errors are detected preventing the masking of non-correctable errors. 2) To enable the DCAM to assert TEA* (Transfer Error Acknowledge) to the processor on read cycles.

The ERLOG* signal is used to latch external information about an error that the ECDM detects such as the address of the error in the BusSwitch PAL register. The ERLOG* signal is connected to the BusSwitch PALOG* input pin. Assertion of this signal stops the BusSwitch from updating the address in the PAL register until ERLOG* is negated. Therefore, the PAL register contains the address of an ECDM logged error that caused ERLOG* to be asserted. The BusSwitch is also capable of generating an interrupt from the ERLOG* signal which enables the system software to be warned of existing correctable errors before they become non-correctable error faults. For additional information on the BusSwitch PAL register and PALOG* signal pin refer to the *Processor Address Log Register - PAL* section of the *BusSwitch* chapter.

ECDM Syndrome Decode

The ECDM detects and corrects errors in the data from a memory system by latching in the 64-bit data word and 8-bit check word and presenting all of this information to a parity tree. This parity tree is similar to the parity tree used to generate the check bits. However, the output from this parity tree is the syndrome bits that are decoded to determine the type of error and it's location if it is a single-bit error. This syndrome code is decoded by logic inside the ECDM to invert a single-bit error or flag a multiple-bit error. Each code, (S7-S0) listed in the syndrome decode chart uniquely identifies one of the data bits in error. A syndrome code of all zeros indicates no error. A single 1 in the syndrome code indicates that the corresponding check-bit is in error. All other variations of syndrome pattern indicate a non-correctable error. If error logging is selected, the syndrome code is latched on the occurrence of errors by error logging circuits in the ECDM; providing system diagnostic data. Refer to Figure 6-3.

DATA	SYNDROME CODE								
ERROR	S7	S6	S5	S4	S3	S2	S1	S0	HEX
D15	0	0	1	0	1	0	0	1	29
D14	0	0	1	1	0	0	0	1	31
D13	1	0	1	1	0	0	0	0	B0
D12	1	0	1	0	1	0	0	0	A8
D11	1	0	1	0	0	1	1	1	A7
D10	0	1	1	1	0	0	0	0	70
D9	0	1	1	0	1	0	0	0	68
D8	0	1	1	0	0	1	0	0	64
RD									
D7	1	0	0	1	0	1	0	0	94
D6	1	0	0	1	1	0	0	0	98
D5	0	1	0	1	1	0	0	0	58
D4	0	1	0	1	0	1	0	0	54
D3	1	1	0	1	0	0	1	1	D3
D2	0	0	1	1	1	0	0	0	38
D1	0	0	1	1	0	1	0	0	34
D0	0	0	1	1	0	0	1	0	32
D15	1	0	1	0	0	1	0	0	A4
D14	1	1	0	0	0	1	0	0	C4
D13	1	1	0	0	0	0	1	0	C2
D12	1	0	1	0	0	0	1	0	A2
D11	1	0	0	1	1	1	1	0	9E
D10	1	1	0	0	0	0	0	1	C1
D9	1	0	1	0	0	0	0	1	A1
D8	1	0	0	1	0	0	0	1	91
RC									
D7	0	1	0	1	0	0	1	0	52
D6	0	1	1	0	0	0	1	0	62
D5	0	1	1	0	0	0	0	1	61
D4	0	1	0	1	0	0	0	1	51
D3	0	1	0	0	1	1	1	1	4F
D2	1	1	1	0	0	0	0	0	E0
D1	1	1	0	1	0	0	0	0	D0
D0	1	1	0	0	1	0	0	0	C8

DATA	SYNDROME CODE								
ERROR	S7	S6	S5	S4	S3	S2	S1	S0	HEX
D15	1	0	0	1	0	0	1	0	92
D14	0	0	0	1	0	0	1	0	13
D13	0	0	0	0	1	0	1	1	0B
D12	1	0	0	0	1	0	1	0	8A
D11	0	1	1	1	1	0	1	0	7A
D10	0	0	0	0	0	1	1	1	07
D9	1	0	0	0	0	1	1	0	86
D8	0	1	0	0	0	1	1	0	46
RB									
D7	0	1	0	0	1	0	0	1	49
D6	1	0	0	0	1	0	0	1	89
D5	1	0	0	0	0	1	0	1	85
D4	0	1	0	0	0	1	0	1	45
D3	0	0	1	1	1	1	0	1	3D
D2	1	0	0	0	0	0	1	1	83
D1	0	1	0	0	0	0	1	1	43
D0	0	0	1	0	0	0	1	1	23
D15	0	1	0	0	1	0	1	0	4A
D14	0	1	0	0	1	1	0	0	4C
D13	0	0	1	0	1	1	0	0	2C
D12	0	0	1	0	1	0	1	0	2A
D11	1	1	1	0	1	0	0	1	E9
D10	0	0	0	1	1	1	0	0	1C
D9	0	0	0	1	1	0	1	0	1A
D8	0	0	0	1	1	0	0	1	19
RA									
D7	0	0	1	0	0	1	0	1	25
D6	0	0	1	0	0	1	1	0	26
D5	0	0	0	1	0	1	1	0	16
D4	0	0	0	1	0	1	0	1	15
D3	1	1	1	1	0	1	0	0	F4
D2	0	0	0	0	1	1	1	0	0E
D1	0	0	0	0	1	1	0	1	0D
D0	1	0	0	0	1	1	0	0	8C
C7	1	0	0	0	0	0	0	0	80
C6	0	1	0	0	0	0	0	0	40
C5	0	0	1	0	0	0	0	0	20
C4	0	0	0	1	0	0	0	0	10
C3	0	0	0	0	1	0	0	0	08
C2	0	0	0	0	0	1	0	0	04
C1	0	0	0	0	0	0	1	0	02
C0	0	0	0	0	0	0	0	1	01

Figure 6-3. ECDM Syndrome Decode

ECDM Configuration in a 64-Bit System

The ECDM is designed to be used in a 64-bit data bus system. This requires four ECDM devices that latch 64 bits of data each. This configuration equates to a 256-bit memory data bus. A major advantage of this configuration is that by-four memory devices may be used for the data and check-bits. Since only one of the four bits from each by-four memory device goes to one of each of the ECDM devices, a failure of an entire memory device is seen as a single-bit error by each ECDM device. The following diagram shows the four ECDM devices and all of the by-four memory devices required to implement this.

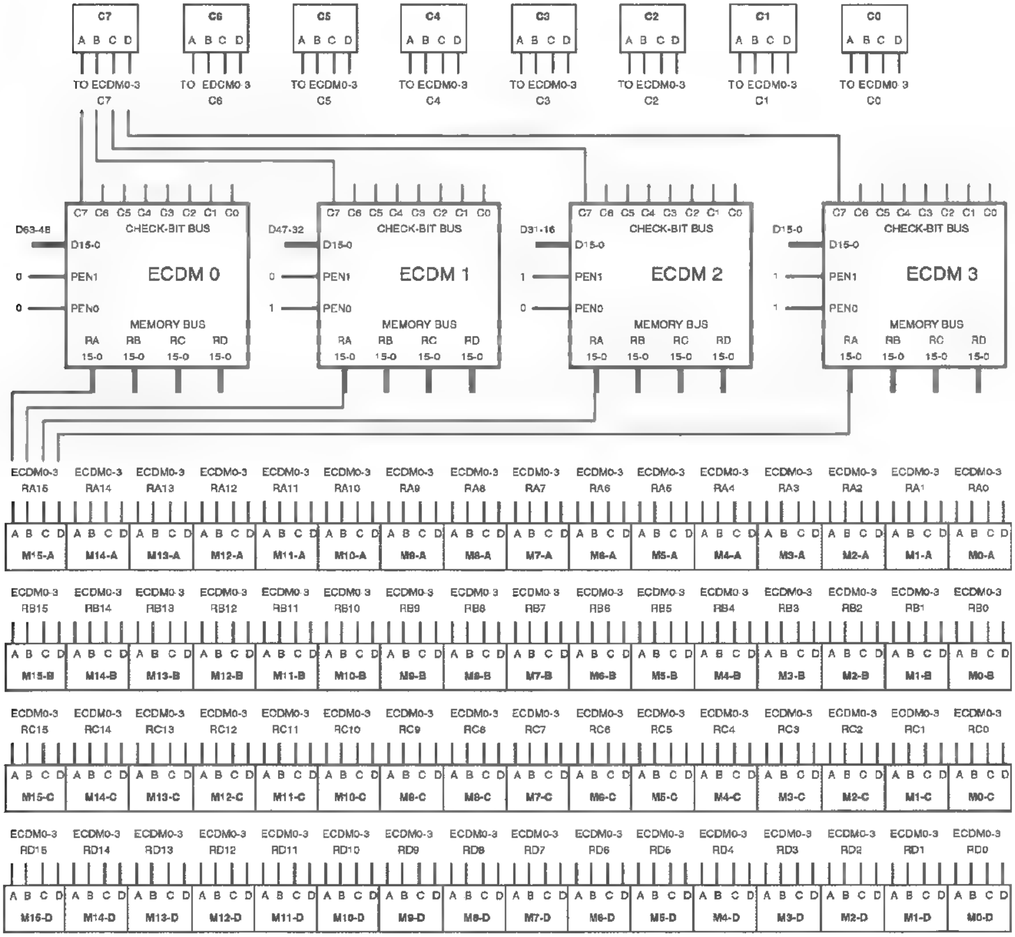

C7 A B C D TO ECDM0-3 C7
C6 A B C D TO EDCM0-3 C6
C5 A B C D TO ECDM0-3 C5
C4 A B C D TO ECDM0-3 C4
C3 A B C D TO ECDM0-3 C3
C2 A B C D TO ECDM0-3 C2
C1 A B C D TO ECDM0-3 C1
C0 A B C D TO ECDM0-3 C0
D63-48
0 PEN1
0 PEN0
C7 C6 C5 C4 C3 C2 C1 C0 CHECK-BIT BUS
D15-0
ECDM 0
MEMORY BUS RA 15-0 RB 15-0 RC 15-0 RD 15-0
D47-32
0 PEN1
1 PEN0
ECDM 1
D31-16
1 PEN1
0 PEN0
ECDM 2
D15-0
1 PEN1
1 PEN0
ECDM 3
ECDM0-3 RA15 RA14 RA13 RA12 RA11 RA10 RA9 RA8 RA7 RA6 RA5 RA4 RA3 RA2 RA1 RA0
A B C D M15-A M14-A M13-A M12-A M11-A M10-A M9-A M8-A M7-A M6-A M5-A M4-A M3-A M2-A M1-A M0-A
ECDM0-3 RB15 RB14 RB13 RB12 RB11 RB10 RB9 RB8 RB7 RB6 RB5 RB4 RB3 RB2 RB1 RB0
A B C D M15-B M14-B M13-B M12-B M11-B M10-B M9-B M8-B M7-B M6-B M5-B M4-B M3-B M2-B M1-B M0-B
ECDM0-3 RC15 RC14 RC13 RC12 RC11 RC10 RC9 RC8 RC7 RC6 RC5 RC4 RC3 RC2 RC1 RC0
A B C D M15-C M14-C M13-C M12-C M11-C M10-C M9-C M8-C M7-C M6-C M5-C M4-C M3-C M2-C M1-C M0-C
ECDM0-3 RD15 RD14 RD13 RD12 RD11 RD10 RD9 RD8 RD7 RD6 RD5 RD4 RD3 RD2 RD1 RD0
A B C D M15-D M14-D M13-D M12-D M11-D M10-D M9-D M8-D M7-D M6-D M5-D M4-D M3-D M2-D M1-D M0-D

Figure 6-4. 64-Bit ECDM Memory Configuration

EDCM I^{2}C Bus Programming Reference

The ECDM/DCAM memory ASICs provide programmable control of the memory sub-system through a combination of registers and an integrated circuit communications interface called the I^{2}C serial bus. For I^{2}C bus programming, the ECDM is the master and the DCAM and serial EEPROM's are slaves. This section will provide examples of the different types of transfers that are used to write to or read from I^{2}C bus slave devices controlled by the ECDM I^{2}C bus interface. The DCAM is a dynamic memory controller and address multiplexer that controls memory timing functions such as refresh and address functions such as address decoding. To minimize pin-count, the DCAM does not have a data interface. All registers within the DCAM are programmed by using the two pin serial I^{2}C bus interface. The DCAM is a slave device on the I^{2}C bus that is controlled by the I^{2}C bus master interface contained in the ECDM. Since the ECDM is designed to perform the data functions for a memory sub-system, it contains the programmable data registers required to change operating modes for either the ECDM or DCAM ASICs.

The ECDM does not have a complete address interface. It has one pin, CSRCS* (Control and Status Register Chip Select), that is connected to the DCAM which asserts CSRCS* when the memory system control and status registers are being accessed by the microprocessor. The DCAM provides this address selection by comparing the current bus address to the value of internal registers. The ECDM allows access to the DCAM internal registers through the I^{2}C bus interface. The I^{2}C bus interface may also be connected to a serial EEPROM and to other ECDM devices. However, only one ECDM I^{2}C bus interface is required to transmit data in the memory sub-system. The I^{2}C bus interfaces contained in other ECDM's of the same memory sub-system are redundant.

Figure 6-5 provides an overview of the ECDM/DCAM memory sub-system CSR architecture.

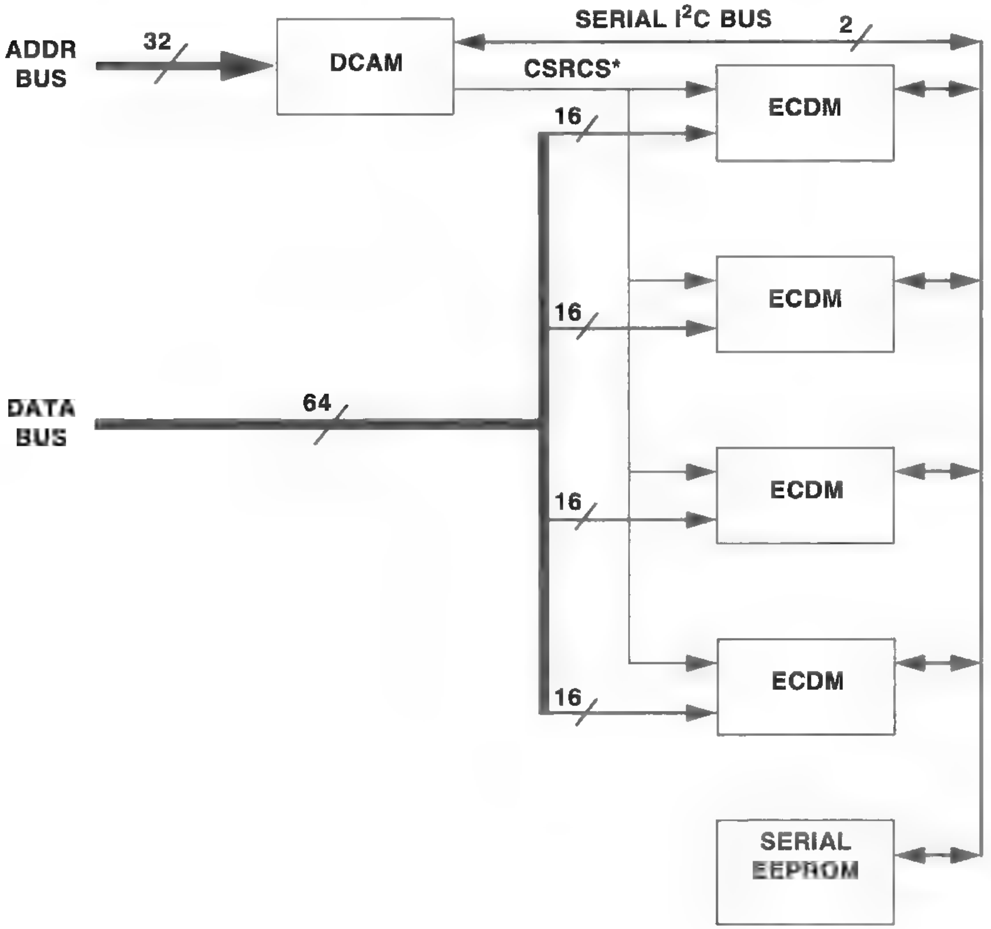


Figure 6-5. Memory Sub-System CSR Architecture

ECDM/DCAM I^{2}C Bus Operation

The ECDM I^{2}C bus transmits data one byte at a time in a serial fashion. Three registers are required to perform the I^{2}C bus data transfer operations. These are the control register (I2CON), the status register (I2STAT), and a data register (I2DATA).

The I2STAT register contains the ICOMP bit. This bit indicates that the ECDM I^{2}C bus controller is ready to perform an operation. I^{2}C bus operations must never be initiated until ICOMP is set. Therefore, the first step in the program sequence should be to test the ICOMP bit for the operation-complete status. The next step is to initiate a START sequence. The START sequence tells I^{2}C bus slaves that the next I^{2}C bus serial data byte contains device address and R/W (read/write) information. The START sequence, device address, and R/W bit are transmitted by first setting the START bit in the I2CON register and then writing the device address (bits 7-1) and R/W bit (bit 0) to the I2DATA register. Nothing will take place on the I^{2}C bus until the I2DATA register has been loaded. For this example, the R/W bit is set to zero indicating a I^{2}C bus write operation. The I2STAT register must now be polled to test the ICOMP and ILRB (Last Received Bit) bits. The ICOMP bit will automatically clear with the write cycle to the I2DATA register. At this time, the START sequence begins. The ICOMP bit becomes set when the device address and R/W bit has been transmitted. The ILRB bit provides status as to whether or not a slave device acknowledged the device address. If so, ILRB will be clear and otherwise set. If ILRB is set following a start sequence, then a fault has occurred. The addressed device is not present on the I^{2}C bus.

With the successful transmission of the device address, the next data value loaded into the I2DATA register will be transmitted to the slave device. This data becomes a pointer to the specific byte address within the slave device. Again, ICOMP and ILRB must be tested for proper response.

The I^{2}C bus slave device and data register location have now been selected. The next data loaded into the I2DATA register will be transmitted to the selected data register within the I^{2}C slave device.

When the desired operations are complete, the STOP sequence must be transmitted to the slave devices. The STOP sequence initiates a programming cycle for serial EEPROMs and also relinquishes the current master's possession of the I^{2}C bus.

The serial EEPROMs and DCAM have the ability for block write or read operations (the number of bytes that are permitted to be transmitted in a block vary with the device). Once the device is addressed (device address and data location transmitted), the slave device automatically increments its internal data pointer after an access has occurred to the register at the current register pointer. Therefore, with I^{2}Cbus devices various transmission modes are possible.

These modes are as follows:

BYTE WRITE A start sequence followed by the first byte which contains the device address and R/W bit is transmitted (R/W=0). The slave device responds with acknowledge. The master transmits the register location pointer byte. The slave responds with acknowledge. The master transmits the data to be loaded into the slave location. The slave will again acknowledge the transfer. The master initiates the STOP sequence.

EXAMPLE

```
TEST I2STAT FOR ICOMP          *Check for interface readiness
LOAD 05 TO ECDM I2CON          *Enables I2C bus and START bit
LOAD C0 TO ECDM I2DATA         *Addr. DCAM, set R/W to write
TEST I2STAT FOR ICOMP          *Check for operation complete
TEST I2STAT FOR ILRB           *Check for slave ack. (zero)
LOAD 05 TO ECDM I2DATA         *Point to DCAM refresh register
TEST I2STAT FOR ICOMP          *Check for operation complete
TEST I2STAT FOR ILRB           *Check for slave ack. (zero)
LOAD 34 TO ECDM I2DATA         *Change memory refresh period
TEST I2STAT FOR ICOMP          *Check for operation complete
TEST I2STAT FOR ILRB           *Check for slave ack. (zero)
LOAD 09 TO ECDM I2CON          *Initiate STOP sequence
TEST I2STAT FOR ICOMP          *Check for operation complete
```

BLK WRITE The block write begins in the same manner as the byte write. However, instead of transmitting the STOP sequence after the data transmission, the master writes another byte of data to the I2DATA register that is transmitted to the next sequential location in the I^2C slave device. The block write is terminated when the master initiates a STOP sequence. The DCAM has the ability to have all of its programmable registers modified in one block. Serial EEPROMs will typically have a limit to the number of locations modified before a STOP sequence must occur to initiate the internal EEPROM programming cycle.

EXAMPLE

```
TEST I2STAT FOR ICOMP          *Check for interface readiness
LOAD 05 TO ECDM I2CON          *Enables I2C bus and START bit
LOAD A0 TO ECDM I2DATA         *Addr. EEPROM, set R/W to write
TEST I2STAT FOR ICOMP          *Check for operation complete
TEST I2STAT FOR ILRB           *Check for slave ack. (zero)
LOAD 00 TO ECDM I2DATA         *Point to first data register
TEST I2STAT FOR ICOMP          *Check for operation complete
TEST I2STAT FOR ILRB           *Check for slave ack. (zero)
LOAD 55 TO ECDM I2DATA         *Change first data location
TEST I2STAT FOR ICOMP          *Check for operation complete
TEST I2STAT FOR ILRB           *Check for slave ack. (zero)
LOAD AA TO ECDM I2DATA         *Change second data location
TEST I2STAT FOR ICOMP          *Check for operation complete
TEST I2STAT FOR ILRB           *Check for slave ack. (zero)
LOAD FF TO ECDM I2DATA         *Change third data location
TEST I2STAT FOR ICOMP          *Check for operation complete
TEST I2STAT FOR ILRB           *Check for slave ack. (zero)
LOAD 00 TO ECDM I2DATA         *Change forth data location
TEST I2STAT FOR ICOMP          *Check for operation complete
TEST I2STAT FOR ILRB           *Check for slave ack. (zero)
LOAD 09 TO ECDM I2CON          *Initiate STOP sequence
TEST I2STAT FOR ICOMP          *Check for operation complete
```

BYTE READ The byte read begins in the same manner as the byte write. The START sequence, device address and R/W bit (R/W=0) must be transmitted to the slave. The pointer byte is transmitted as well. At this point, the slave device is still in a write mode. Therefore, another start sequence, device address and R/W bit (R/W=1) must be sent to the slave to change the mode to read. The current mode (read or write) can be determined by examining the ITX bit in the ECDM I2STAT register. After the mode has been changed to read, the master writes a dummy value to the I2DATA register (data=don't care). This causes the ECDM I^{2}C controller to initiate a read transmission from the slave device. When the master has received the byte of data (indicated by the IRCV bit in the I2STAT register), the system software may then read the data by performing a read of the I2DATA register. The master does not acknowledge the read data for a single byte transmission on the I^{2}C bus. The master completes the transmission by initiating a STOP sequence.

EXAMPLE

```
TEST I2STAT FOR ICOMP          *Check for interface readiness
LOAD 05 TO ECDM I2CON          *Enables I²C bus and START bit
LOAD C0 TO ECDM I2DATA         *Addr. DCAM, set R/W to write
TEST I2STAT FOR ICOMP          *Check for operation complete
TEST I2STAT FOR ILRB           *Check for slave ack. (zero)
LOAD 00 TO ECDM I2DATA         *Point to DCAM ID register
TEST I2STAT FOR ICOMP          *Check for operation complete
TEST I2STAT FOR ILRB           *Check for slave ack. (zero)
LOAD 07 TO ECDM I2CON          *Enbl. I²C bus, ACK and START bit
LOAD C1 TO ECDM I2DATA         *Addr. DCAM, set R/W to read
TEST I2STAT FOR ICOMP          *Check for operation complete
LOAD 01 TO ECDM I2CON          *Disable read acknowledge
LOAD XX TO ECDM I2DATA         *Initiate read transmission
TEST I2STAT FOR ICOMP          *Check for operation complete
READ I2DATA REGISTER           *DCAM ID value read
LOAD 09 TO ECDM I2CON          *Initiate STOP sequence
TEST I2STAT FOR ICOMP          *Check for operation complete
```

BLK READ — The block read begins in the same manner as the byte read. However, the master acknowledges the I^2C bus read data transmission until it no longer desires the block read to continue. As long as the master acknowledges the read transmission, the slave will be prepared to deliver the next sequential byte of data.

EXAMPLE

TEST I2STAT FOR ICOMP	*Check for interface readiness
LOAD 05 TO ECDM I2CON	*Enable I^2C bus and START bit
LOAD C0 TO ECDM I2DATA	*Addr. DCAM, set R/W to write
TEST I2STAT FOR ICOMP	*Check for operation complete
TEST I2STAT FOR ILRB	*Check for slave ack. (zero)
LOAD 00 TO ECDM I2DATA	*Point to DCAM ID register
TEST I2STAT FOR ICOMP	*Check for operation complete
TEST I2STAT FOR ILRB	*Check for slave ack. (zero)
LOAD 07 TO ECDM I2CON	*Enbl. I^2C bus, ACK and START bit
LOAD C1 TO ECDM I2DATA	*Addr. DCAM, set R/W to read
TEST I2STAT FOR ICOMP	*Check for operation complete
LOAD XX TO ECDM I2DATA	*Initiate read transmission
TEST I2STAT FOR ICOMP	*Check for operation complete
READ I2DATA REGISTER	*First byte read complete
LOAD XX TO ECDM I2DATA	*Initiate read transmission
TEST I2STAT FOR ICOMP	*Check for operation complete
READ I2DATA REGISTER	*Second byte read complete
LOAD XX TO ECDM I2DATA	*Initiate read transmission
TEST I2STAT FOR ICOMP	*Check for operation complete
READ I2DATA REGISTER	*Third byte read complete
LOAD 01 TO ECDM I2CON	*Disable read acknowledge
LOAD XX TO ECDM I2DATA	*Initiate read transmission
TEST I2STAT FOR ICOMP	*Check for operation complete
READ I2DATA REGISTER	*Forth byte read complete
LOAD 09 TO ECDM I2CON	*Initiate STOP sequence
TEST I2STAT FOR ICOMP	*Check for operation complete

ECDM Register Map in a 64-Bit System

As stated earlier, the MVME197 uses four ECDM ASICs and one DCAM ASIC to perform the memory sub-system function. In this configuration, each ECDM resides on 16 bits of the MC88110 data bus. All programmable registers for the sub-system are contained within the ECDM devices. The DCAM registers are programmed via the I^{2}C bus interface in the ECDM that connects to the DCAM through a serial interface.

Table 6-1 illustrates all of the registers in the MVME197 memory sub-system. The address offset of each register with respect to the memory CSR base address is given along with the name for each register. The bottom part of the table provides the portion of the MC88110 data bus where each ECDM resides.

ECDM Control and Status Registers

The ECDM has an internal CSR (Control and Status Register) that provides configuration and diagnostic functions as well as status information. The CSR is decoded externally and is selected by asserting the CSRCS* (CSR Chip Select) signal. The CSR controls the ECDM's diagnostic and operational modes. In addition, the CSR holds the error logger status registers and I^{2}C registers. The I^{2}C serial bus is used to transmit data to or from the DCAM or a serial device such as a serial EEPROM. Refer to Figure 6-5 for an overview of the ECDM/DCAM memory sub-system CSR architecture.

CSR Operation

The memory CSR control bits are read/write and the error logger status bits are read only and cleared on read. If the ECDM has asserted the ERLOG* signal, reading the error logger status will cause the ERLOG* signal to be negated. However, since the ERLOG* signal is open drain, all of the ECDM device status registers must be read to determine if they are asserting the ERLOG* signal as well.

The diagnostic functions, such as read/write check-bits, should be used with caution since this mode causes normal memory program and data accesses to be replaced with check-bit information. However, this mode provides a useful method for memory device testing of the check-bits. The I^{2}C control bits are read/write and the I^{2}C status bits are read only. The ECDM I^{2}C interface is master only. Therefore, some of the bits defined for slave operation are not used.

The following sections describe the ECDM control and status registers.

Table 6-1. ECDM CSR Register Memory Map

Sub-System Memory CSR Base Address = $FFF01000

Offset/Register:

ECDM0		ECDM1		ECDM2		ECDM3	
ADDR/REGISTER	ADDR/REGISTER	ADDR/REGISTER	ADDR/REGISTER	ADDR/REGISTER	ADDR/REGISTER	ADDR/REGISTER	ADDR/REGISTER
00 / MEMCON0	01 / ECDMID0	02 / MEMCON1	03 / ECDMID1	04 / MEMCON2	05 / ECDMID2	06 / MEMCON3	07 / ECDMID3
08 / SYNSTAT0	09 / ERSTAT0	0A / SYNSTAT1	0B / ERSTAT1	0C / SYNSTAT2	0D / ERSTAT2	0E / SYNSTAT3	0F / ERSTAT3
10 / I2CON0	11 / I2STAT0	12 / I2CON1	13 / I2STAT1	14 / I2CON2	15 / I2STAT2	16 / I2CON3	17 / I2STAT3
18 / I2DATA0	19 / I2ADDR0	1A / I2DATA1	1B / I2ADDR1	1C / I2DATA2	1D / I2ADDR2	1E / I2DATA3	1F / I2ADDR3

D63 D56	D55 D48	D47 D40	D39 D32	D31 D24	D23 D16	D15 D8	D7 D0

ECDM register map of four ECDM devices in a 64-bit system. The byte offset address is shown next to each register.

Memory Control Register - MEMCON

REG	Memory Control Register							
ADR/SIZ	$FFF01000 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	EERO	CLSIZ	NOCOR	INIT	RWCKB	LOG NCER	LOGCER	EERP
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0

The MEMCON register is used to control the ECDM mode of operation.

EERO Enable Error Output. When set, this bit enables the ECDM error indication output signals (NCER* and CER*) to be asserted on the detection of errors. EERO does not affect error logging or the ERLOG* signal. This bit should remain clear until the entire memory subsystem is initialized by writing data to each location. This bit should not be set unless the DCAM has been programmed for TEA* assertion on non-correctable errors. When DCAM programming and initialization are complete, the desired error logging and error reporting option can be selected and this bit can be set.

CLSIZ Cache Line Size. This bit configures the ECDM for the desired cache line size. When set, a cache line size of 8 is selected. When cleared, a cache line size of 4 is selected. This bit must remain clear for proper operation with the MC88110 processor. If the ECDM is used to interface between the MC88410 cache controller and the memory system, then 4 or 8 cache line lengths are valid depending on the system configuration.

NOCOR No Error Correction. When set, this bit prevents the error correction circuitry from affecting the data read from memory but does not affect normal check-bit or syndrome generation. This mode is useful for diagnostic testing of the error correction system.

INIT Initialize Mode Control. When set, this bit forces write data to the memory data busses to be zero. This bit is intended to allow fast initialization of the memory subsystem. It is important to note that the normal read/write operation of the memory system is prevented if this bit is set. Therefore, INIT must be cleared for normal operation.

RWCKB Read/Write Check-bits. When set, this bit enables the data from the 8 check-bits for each ECDM device to be written to and read on data bits D7-D0 of each device. Since the error correction is performed on double-word boundaries, the check-bit data is valid for one byte on each double-word boundary (refer to the *General Description* section of this chapter for a further description of the EDAC configuration and organization). This bit is used during memory diagnostic testing and should remain cleared for normal operation.

LOGNCER Log Non-Correctable Errors. The most recent error will always be logged by the ECDM unless one of the LOGX bits is set (LOGNCER or LOGCER). LOGNCER when set, locks the error logger circuitry on non correctable errors. This bit does not prevent the Non-Correctable Error signal (NCER*) from being asserted. However, when cleared, this bit does prevent the ERLOG* signal from being asserted only for non-correctable errors. This bit is typically set for normal system operation.

LOGCER Log Correctable Errors. The most recent error will always be logged by the ECDM unless one of the LOGX bits (LOGNCER or LOGCER) is set. LOGCER when set, locks the error logger circuitry to on correctable errors. This bit does not prevent the Correctable Error signal (CER*) from being asserted. However, when cleared, this bit prevents the ERLOG* signal from being asserted only for correctable errors. This bit is typically cleared for normal system operation. However, logging correctable error information may be useful for special system diagnostics i.e., in determining pages of memory to avoid or to notify system services of a possible device failure.

EERP Enable Error Reporting. When set, this bit enables the ECDM to assert the ERLOG* signal. EERP does not affect the error logger or the error indication output signals (NCER*, CER*) in the ECDM. This bit should remain clear until the entire memory subsystem is initialized by writing data to each location. When initialization is completed, the desired error logging option can be selected and this bit can be set.

ECDM ID Register - ECDMID

REG	ECDM ID Register							
ADR/SIZ	$FFF01001 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
OPER	R	R	R	R	R	R	R	R
RESET	1	0	0	0	0	0	1	1

The ECDMID register contains the ECDM device ID code h83.

ID7-ID0 Identification 7-0. The ECDMID register provides ID information intended for use by system software to differentiate the ECDM from similar devices on present or future products.

Syndrome Status Register - SYNSTAT

REG	Syndrome Status Register							
ADR/SIZ	$FFF01008 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	S7	S6	S5	S4	S3	S2	S1	S0
OPER	R	R	R	R	R	R	R	R
RESET	0	0	0	0	0	0	0	0

The SYNSTAT register contains the syndrome code that is latched when the ERLOG* signal is asserted. The SYNSTAT and ERSTAT registers perform the error logger function in the ECDM.

S7-0 Syndrome Code 7-0. The SYNSTAT register contains the coded information regarding data bit failures. For correctable errors, the syndrome code may be used to determine the bit that failed (see Figure 6-3). The SYNSTAT register latches syndrome information when either the RDLA1* or RDLA0* signal pins are asserted. If the ECDM detects an error and asserts the ERLOG* signal, error logging is suspended until the error status is read. Once ERLOG* is asserted, none of the ECDM devices will log errors whether or not ERLOG* is internally or externally generated. This byte contains valid syndrome information if the CER or NCER status bit in the ERSTAT register are set. If NCER is set and the SYNSTAT is non-zero, the SYNSTAT register contains the syndrome code of the last correctable error before the non-correctable error occurred. This allows some history of the memory fault that may have led to the non-correctable error.

Reading the ERSTAT register clears both the SYNSTAT and ERSTAT registers and also negates the ERLOG* signal for this ECDM device. For this reason, the ERLOG* signal is well suited for generating an interrupt. Once an error has been logged by any of the ECDM devices in a system, another error will not be logged until the ERSTAT register of the ECDM that is asserting ERLOG* is read. It is possible for multiple ECDM devices to log errors, even of different type, at the same time. Therefore, if a memory error exception occurs, the error loggers of all the ECDM devices should be read. If the ERLOG* signal is used for latching other information external to the ECDM (such as address information of the error in the BusSwitch) these other registers must be read before the ERSTAT register is read.

Error Type Status Register - ERSTAT

REG	Error Type Status Register							
ADR/SIZ	$FFF01009 (6 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	NCER	CER	INIT STAT	MODE	STAT410	VERS2	VERS1	VERS0
OPER	RO	RO	RO	RO	RO	RO	RO	RO
RESET	0	0	0	0	0	0	0	1

The ERSTAT register contains error-type status that is latched when the RDLA* signals are asserted. The SYNSTAT and ERSTAT registers perform the error logger function in the ECDM (refer to the description of the error logger function in the section above).

Note **The ERSTAT register is cleared when read (except INITSTAT).**

NCER Non-Correctable Error Status. When set, this bit indicates that a non-correctable error has occurred. This status bit is enabled when LOGNCER in the MEMCON register is set. Syndrome status in the SYNSTAT register is not affected if this bit is set. Therefore, the SYNSTAT register will contain the last single bit error that occurred before the non-correctable error occurred. All error reporting and error logging is disabled if the EERP bit in the MEMCON register is cleared.

CER Correctable Error Status. When set, this bit indicates that a correctable error has occurred. This status bit is enabled when LOGCER in the MEMCON register is set. Syndrome status in the SYNSTAT register indicates the bit in error when this bit is set.

INITSTAT Initialization Status. When set, this bit indicates that the memory sub-system is in the initialization mode. This bit is set when any of the ECDM devices connected to the INIT* pin have asserted INIT*. Therefore, INITSTAT will be set when INIT, in any ECDM MEMCON register, is set. INITSTAT is not cleared when the ERSTAT register is read.

MODE 110/040 Mode Control. This bit changes the way the ECDM decodes the size signals (SIZ1,0). When set, this bit enables the ECDM to be used with a MC68040 type bus interface. When low, the ECDM decodes size is determined by the MC88110 bus interface definition. This bit has two possible reset states depending on the hardware configuration. If the TBST* signal is asserted low coincident with the transition from low to high of the POR* pin, the ECDM will set this bit and come up in the MC68040 mode. If TBST* is negated when POR* transitions low to high, the ECDM will clear this bit and come up in the MC88110 mode. It is not possible to set or clear this bit by any other means.

STAT410 88410 Mode Status. When set, this bit indicates that the ECDM has detected the MC88410 cache presence. This bit is set only by the detection of PTA* low when POR* transition from low to high. This bit is useful for operating system or diagnostic software configuration checking.

VERS2-0 Version Status (bits 2 through 0). VERS2-0 indicate the revision of the ECDM. The ECDM is version is 2 (VERS1 = 1). If the ECDM is updated by a new layout of the chip, the version will increment by 1.

I^{2}C Control Register - I2CON

REG	I^{2}C Control Register							
ADR/SIZ	$FFF01010 (4 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD					ISTOP	ISTART	IACK	IENA
OPER					R/W	R/W	R/W	R/W
RESET					0	0	0	0

The I2CON register is used to control the ECDM I^{2}C serial bus interface. The I^{2}C bus is used to program the DCAM control registers. The DCAM registers may be read, indirectly, over the I^{2}C interface as well. In addition, a general purpose serial EEPROM may be connected to the I^{2}C interface. The EEPROM may hold configuration, address selection, or initialization data for the DCAM. Alternatively, it may be used for general purpose data that must be retained when power is removed from the system.

Note **Bits 7-4 of the I2CON register are reserved for I^{2}C slave mode which is not supported in the ECDM implementation.**

ISTOP Stop Generate. When set, this bit generates a stop sequence on the I^{2}C bus (assuming the ECDM I^{2}C controller is the current I^{2}C bus master) and clears the ICOMP bit in the I2STAT register. The ECDM I^{2}C controller automatically clears the ISTOP bit when the stop sequence has been completed and then sets the ICOMP bit.

ISTART Start Generate. When set, the ECDM I^{2}C controller generates a start sequence on the I^{2}C bus on the next write to the I2DATA register and clears the ICOMP bit in the I2STAT register. The ECDM I^{2}C controller automatically clears the ISTART bit when the start sequence has been completed. The ECDM I^{2}C controller sets the ICOMP bit when the start sequence and I2DATA register contents have been transmitted.

IACK Acknowledge Generate. When set, the ECDM I^{2}C controller generates an acknowledge on the I^{2}C bus during read cycles. This bit is normally set accept on the last byte of a multi-byte read sequence. IACK does not affect I^{2}C bus write cycles.

IENA I^{2}CC Enable. This bit must be set to enable the ECDM I^{2}C interface. If clear, reads and writes to the ECDM I^{2}C registers are ignored.

I^{2}C Status Register - I2STAT

REG	I^{2}C Status Register							
ADR/SIZ	$FFF01011 (7 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	ICOMP	ILRB	IOWN	IBBSY		IRCV	ITX	ILOST
OPER	R	R	R	R		R	R	R
RESET	1	1	0	0		0	0	0

The I2STAT register provides status information regarding the current state of the I^{2}C bus and ECDM I^{2}C controller.

6

Note **Bit 3 of the I2STAT register is reserved for I^{2}C slave mode which is not supported in the ECDM implementation.**

ICOMP I^{2}C Operation Complete. When set, this bit indicates that the previous I^{2}C operation is complete. This bit must be set before any of the I^{2}C registers are modified. ICOMP is cleared when a valid I^{2}C operation is requested. ICOMP is set automatically when the ECDM I^{2}C controller has completed the requested operation.

ILRB Last Received Bit. This bit contains the last bit received by the I^{2}C controller during the last read or write operation (the acknowledge bit). On start sequences or data writes, this bit is cleared if the addressed device responded and set otherwise. On data read operations, this bit is cleared if the ECDM I^{2}C controller acknowledged the read cycle and set otherwise.

IOWN I^{2}C bus Ownership. This bit is set when the ECDM I^{2}C controller is the current bus master. When IOWN is clear a start sequence must be initiated to gain ownership of the I^{2}C bus. If IOWN is clear writes to the I2DATA register, as well as stop generation requests, are ignored. If IOWN is set, data transfers or start sequences may be performed. When I^{2}C bus ownership is no longer required, a stop sequence should be generated to relinquish possession of the I^{2}C bus. If the ECDM I^{2}C controller loses arbitration to another I^{2}C bus master, IOWN is cleared and ILOST is set; meaning that the transfer should be re-run from the start sequence.

IBBSY I^{2}C bus Busy. This bit indicates that another I^{2}C bus master currently owns the I^{2}C bus. However, it is not necessary to wait for this bit to clear before initiating an I^{2}C bus transfer. This bit is set when a start sequence is detected and cleared when a stop sequence is detected on the I^2C bus.

IRCV Received Read Data. This bit is set whenever the ECDM I^2C controller has received a byte of read data from an I^2C bus device. This bit is cleared when the I2DATA register is read.

ITX Transmit Mode. ITX is set when the last bit of the start sequence (the R/W bit) is a one. The last bit of a start sequence is determined by the value of the LSB in the I2DATA register when ISTART is set in the I2CON register and a write to the I2DATA register occurs. If ITX is set, writes to the I2DATA register cause read operations to be performed. If ITX is cleared, writes to the I2DATA register transmit the contents of the I2DATA register to I^2C slave device.

ILOST Arbitration Lost. This bit indicates that the ECDM I^2C controller has lost arbitration to another I^2C bus master. The requested I^2C transfer was aborted and must be re-run from the start sequence.

I^2C Data Register - I2DATA

REG	I^2C Data Register							
ADR/SIZ	$FFF01018 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	I2D7	I2D6	I2D5	I2D4	I2D3	I2D2	I2D1	I2D0
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0	0	0	0	0	0	0	0

I^2C operations are initiated by writes to the I2DATA register except for stop generation. When ITX in the I2STAT register is clear, writes to the I2DATA register transmit the value to the responding I^2C device. If ITX is set, writes to the I2DATA register (value = don't care) initiate a read from the responding I^2C device. The state of the ITX bit is determined by the LSB of the data byte transmitted immediately following a start sequence. Writes to the I2DATA register have no affect on the I^2C bus if the ECDM I^2C controller does not own the bus or has lost the bus to another I^2C master. In these cases, a start sequence must be initiated to gain possession of the I^2C bus.

I2D7-0 I^2C Data. The I2DATA register is the transmit and receive byte for I^2C data transfers. If the I2DATA register is written to when the ISTART bit in the I2CON register is set, a start sequence is generated immediately followed by the transmission of the contents of the I2DATA register (the device address and R/W bit). The value of I2D7-1 is the device address. I2D0 is the R/W bit (0=WRITE, 1=READ). After a start sequence with I2D0=0, writes to the I2DATA register cause the contents to be transmitted to the responding device. After a start sequence with I2D0=1, writes to the I2DATA register (value = don't care) cause the responding device to transmit data to the I2DATA register.

I^{2}C Address Register - I2ADDR

REG	I^{2}C Address Register							
ADR/SIZ	$FFF01019 (8 bits)							
BIT	7	6	5	4	3	2	1	0
FIELD	I2A7	I2A6	I2A5	I2A4	I2A3	I2A2	I2A1	I2A0
OPER	R	R	R	R	R	R	R	R
RESET	0	0	0	0	0	0	0	0

The I2ADDR register is reserved for I^{2}C bus slave mode which is not supported by the ECDM I^{2}C controller.

I2A7-0 I^{2}C Slave Address. The I2ADDR register is used to set the address of the I^{2}C controller when responding to I^{2}C bus transfers as a slave. The ECDM I^{2}C controller does not support slave mode and always returns the value h00 when the I2ADDR register is read.

PRINTER AND SERIAL PORT CONNECTIONS 7

Introduction

This chapter has connection diagrams for the printer port and the four serial ports on the MVME197. These ports are connected to external devices through the MVME712 series of transition modules. The figures showing this are as follows:

Figure Number	Name
7-1	Printer Port with MVME712A
7-2	Printer Port with MVME712M
7-3	Serial Port 1 Configured as DCE
7-4	Serial Port 2 Configured as DCE
7-5	Serial Port 3 Configured as DCE
7-6	Serial Port 4 Configured as DCE
7-7	Serial Port 1 Configured as DTE
7-8	Serial Port 2 Configured as DTE
7-9	Serial Port 3 Configured as DTE
7-10	Serial Port 4 Configured as DTE
7-11	Serial Port 1 with MVME712A
7-12	Serial Port 2 with MVME712A
7-13	Serial Port 3 with MVME712A
7-14	Serial Port 4 with MVME712A

The configuration of the serial ports as Data Terminal Equipment (DTE) or Data Circuit-terminating Equipment (DCE) is accomplished by jumpers on the MVME712 series of transition modules. For more information, refer to the *MVME712-12, MVME712-13, MVME712A, MVME712AM, and MVME712B Transition Module and LCP2 Adapter Board User's Manual* or the *MVME712M Transition Module and P2 Adapter Board User's Manual* for more information.

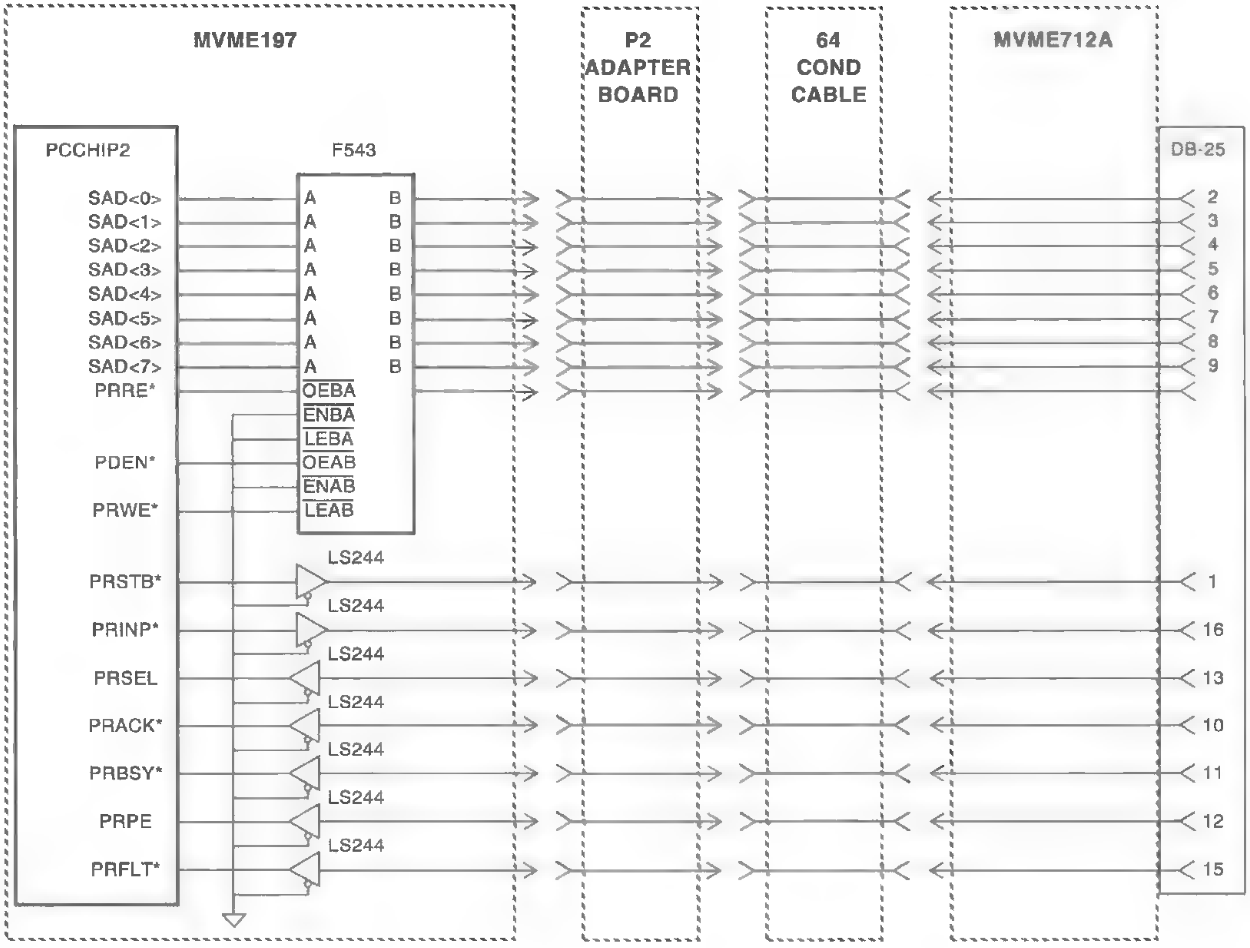


Figure 7-1. Printer Port with MVME712A

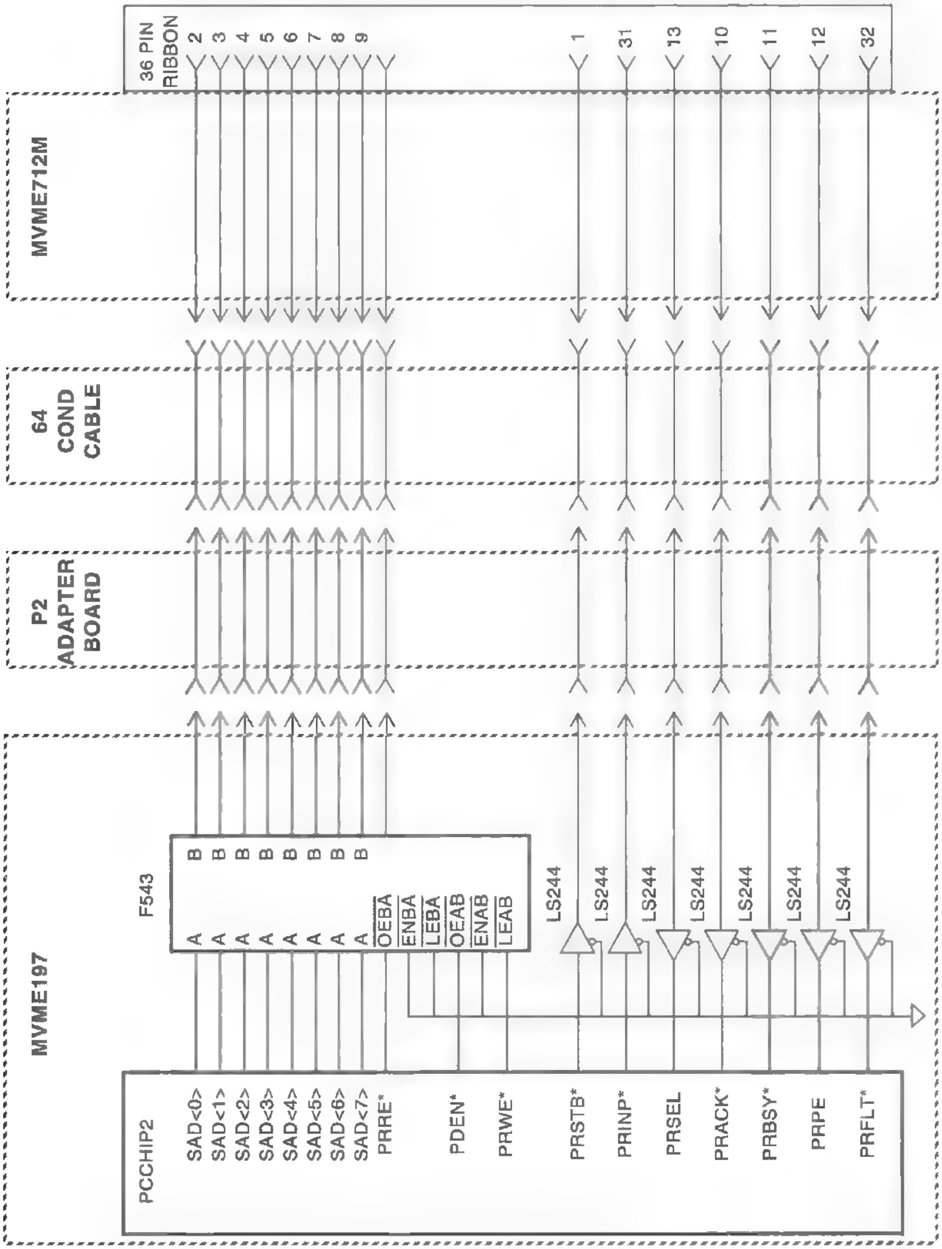


Figure 7-2. Printer Port with MVME712M

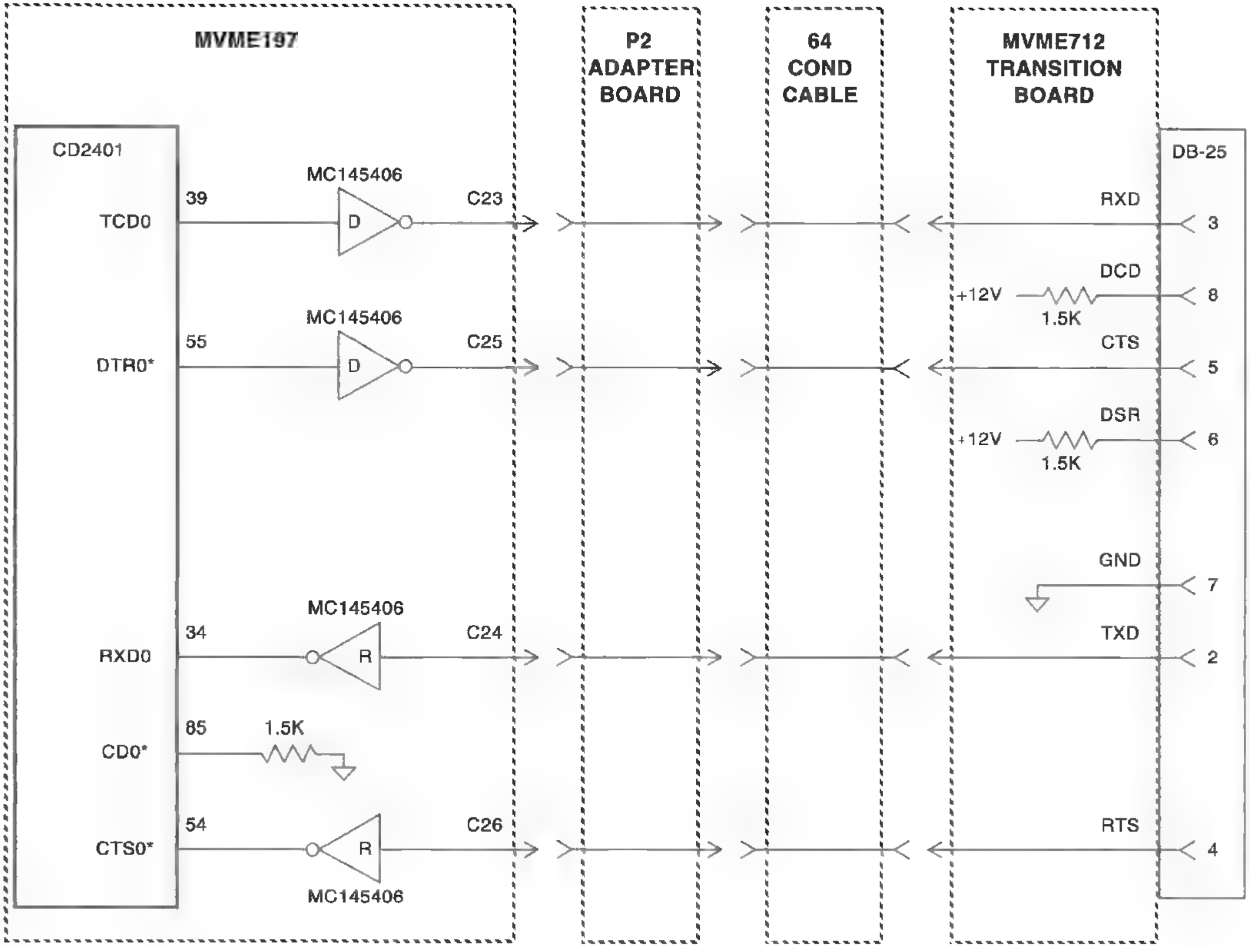


Figure 7-3. Serial Port 1 Configured as DCE

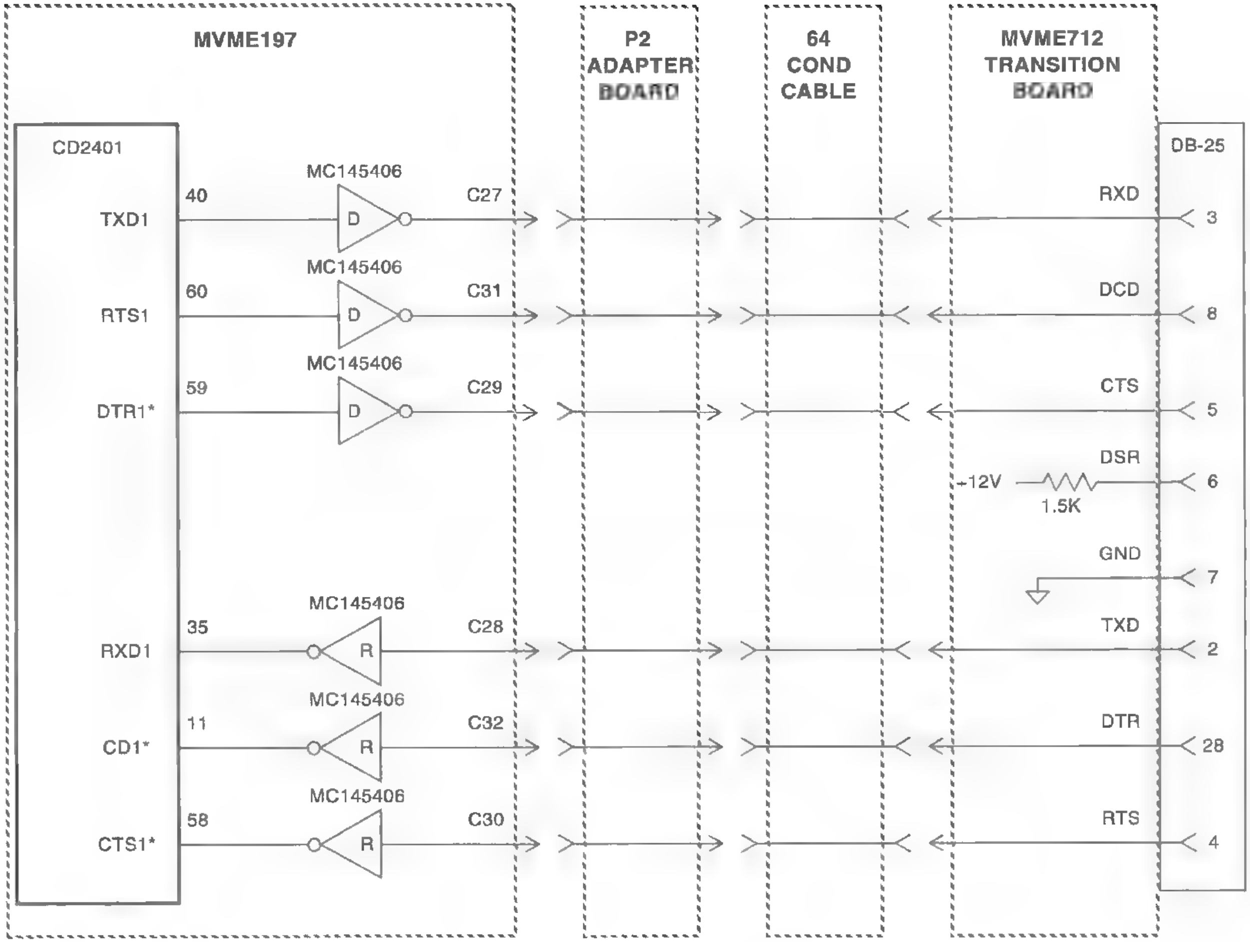


Figure 7-4. Serial Port 2 Configured as DCE

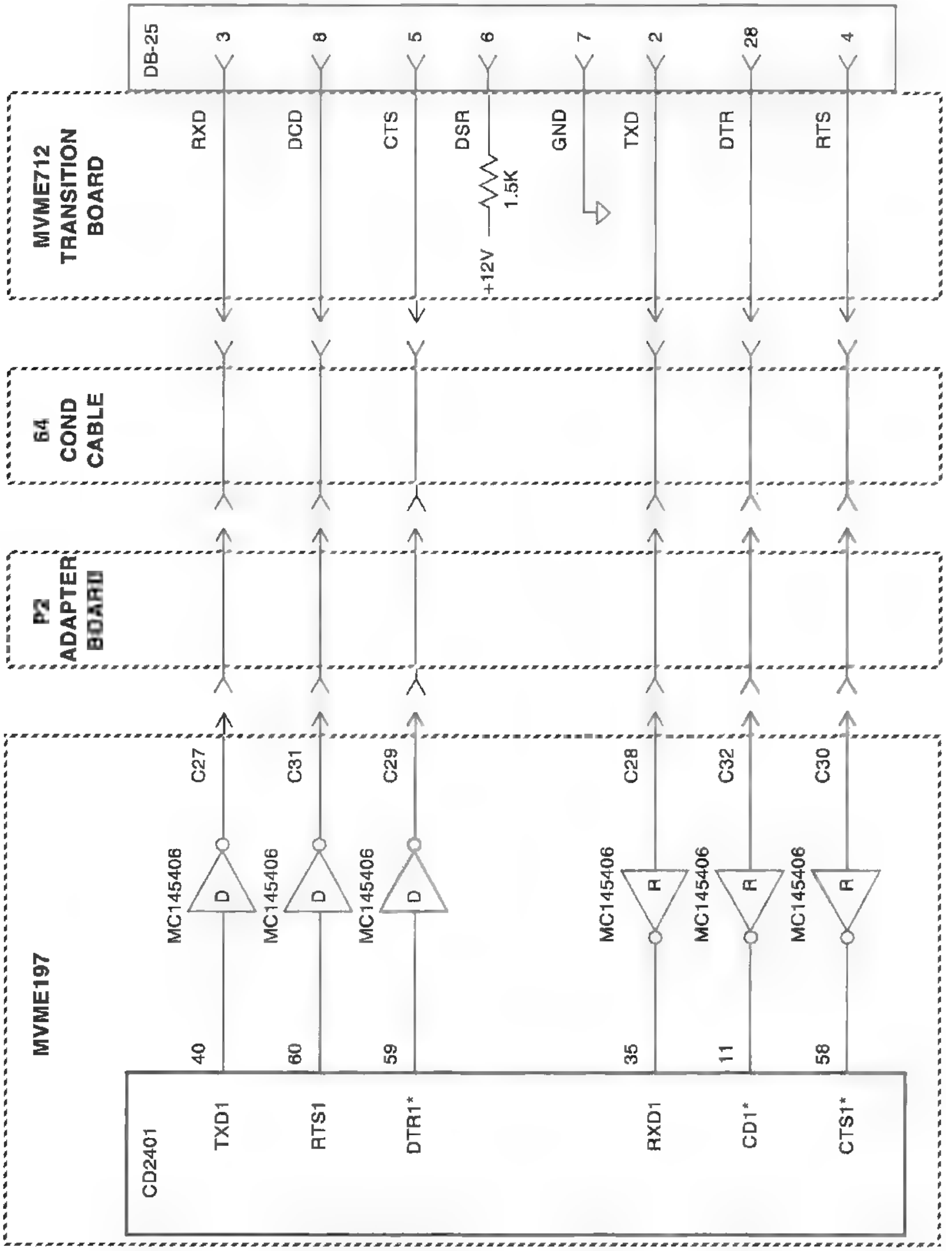


Figure 7-5. Serial Port 2 Configured as DCE

"

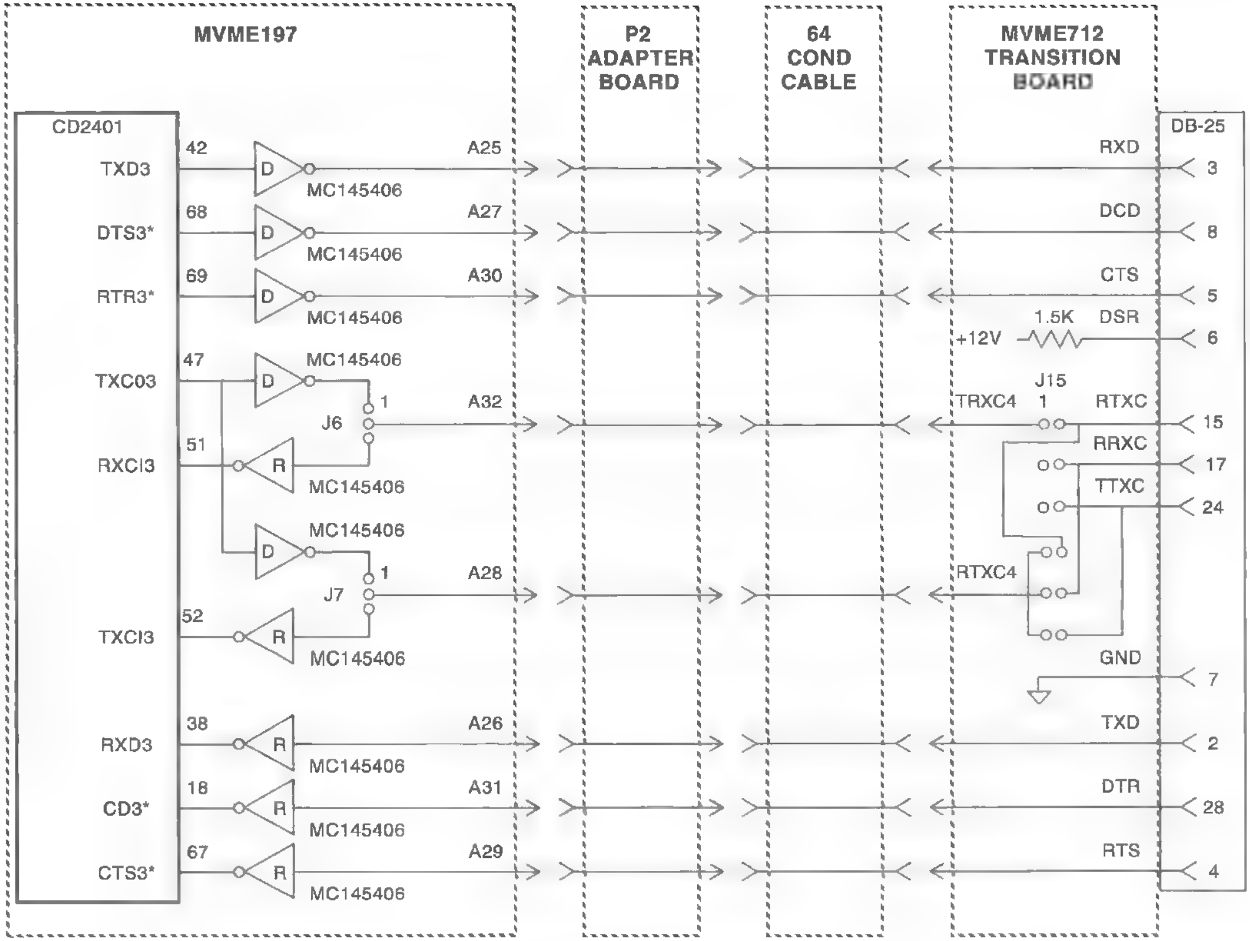


Figure 7-6. Serial Port 4 Configured as DCE

7

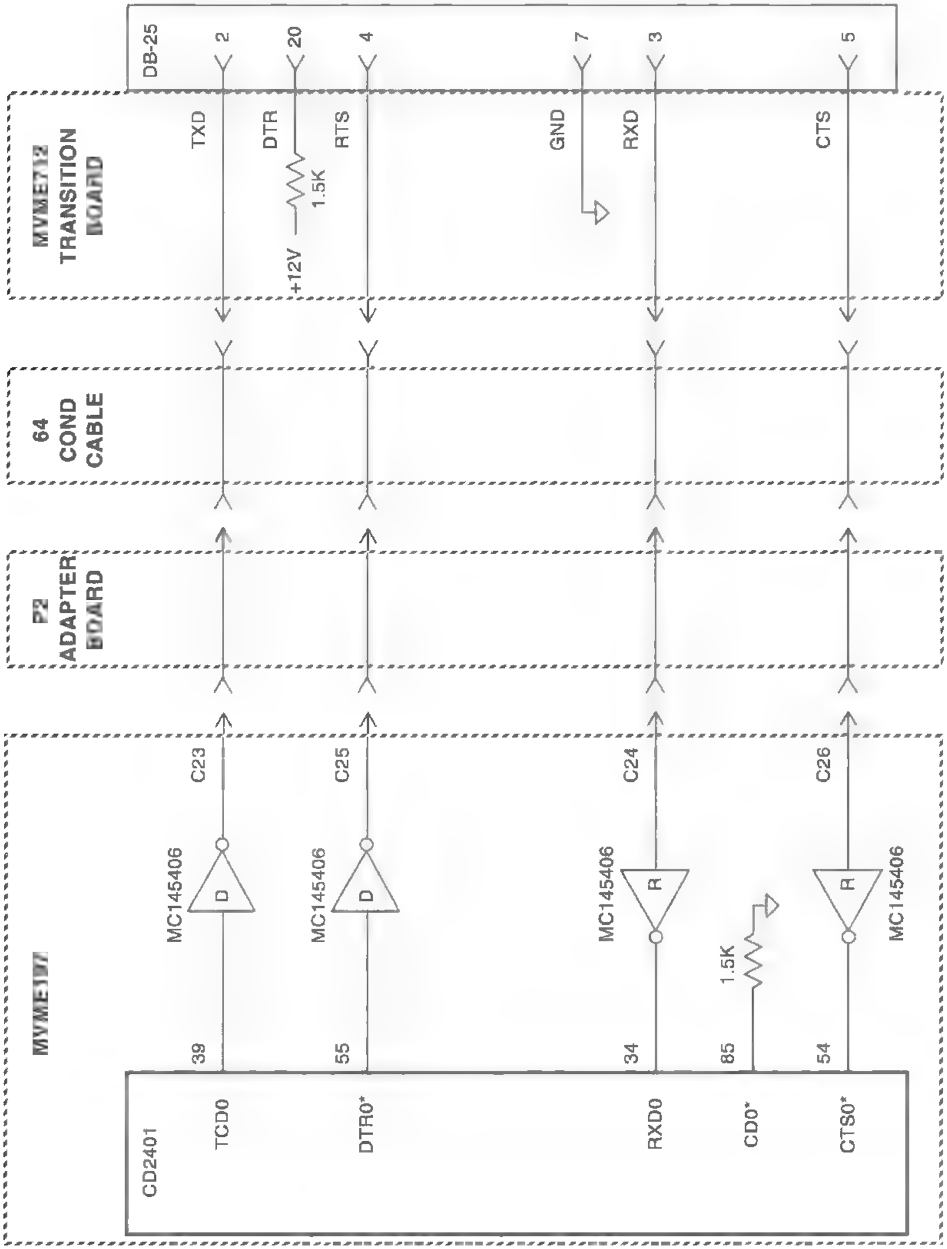


Figure 7-7. Serial Port 1 Configured as DTE

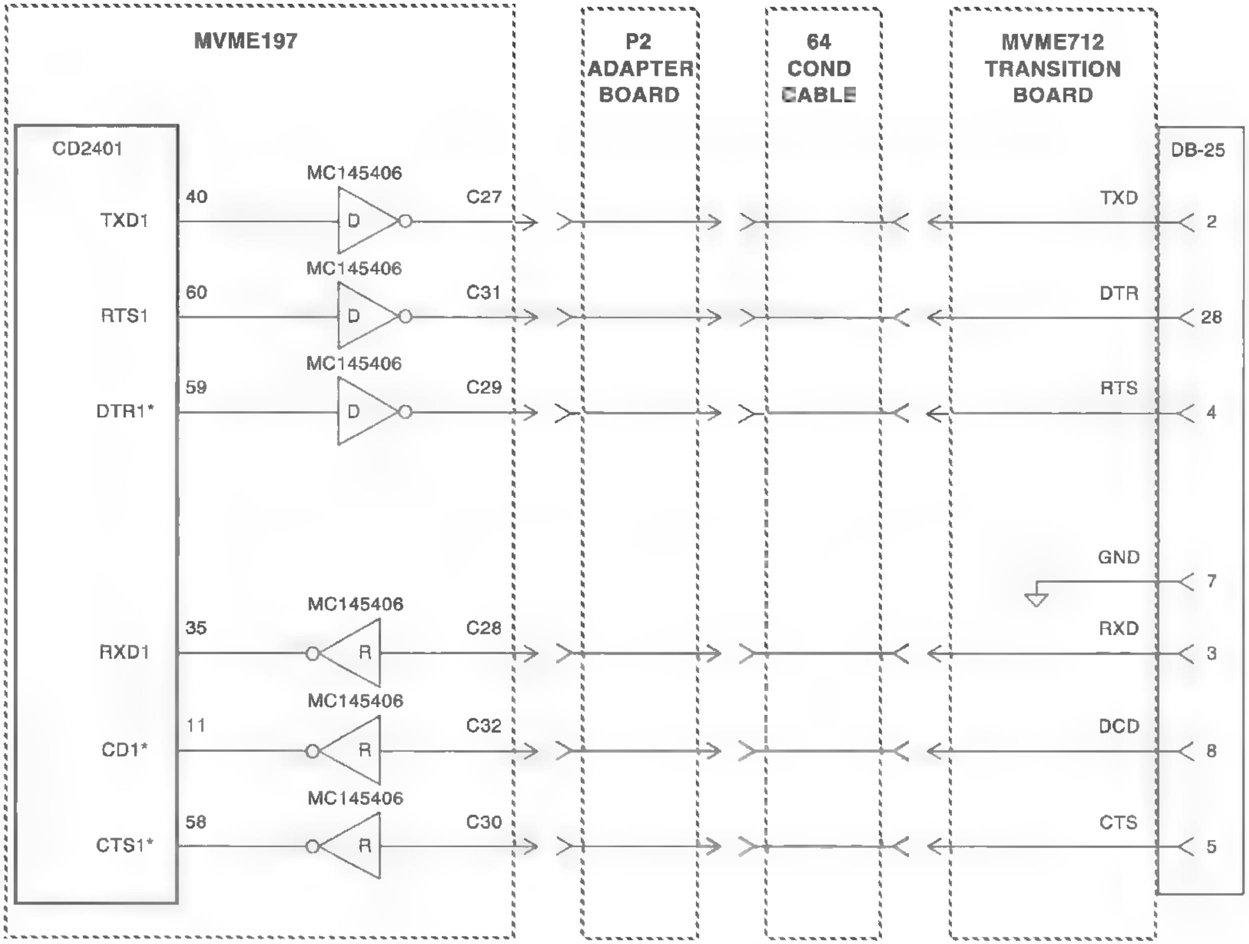


Figure 7-8. Serial Port 2 Configured as DTE

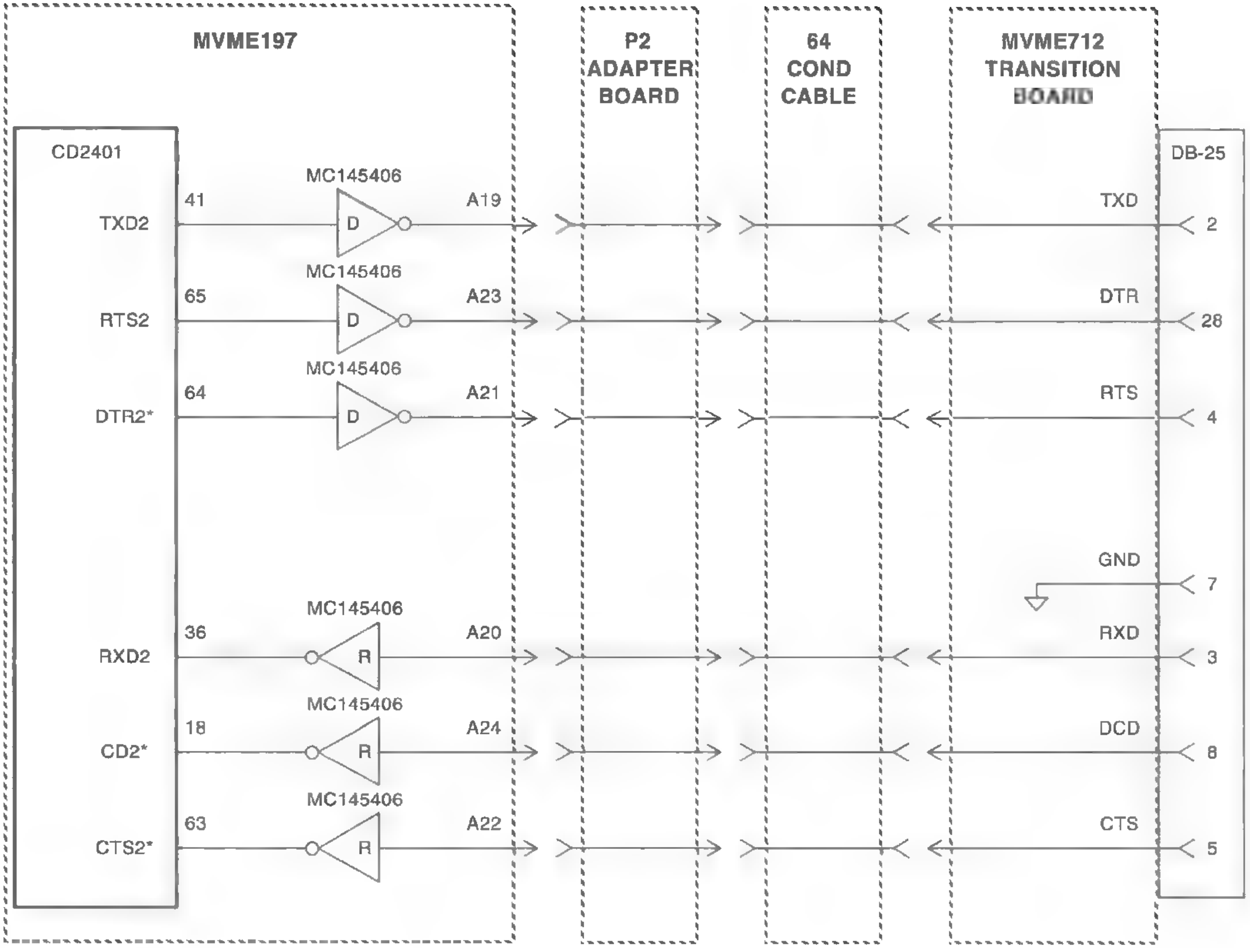


Figure 7-9. Serial Port 3 Configured as DTE

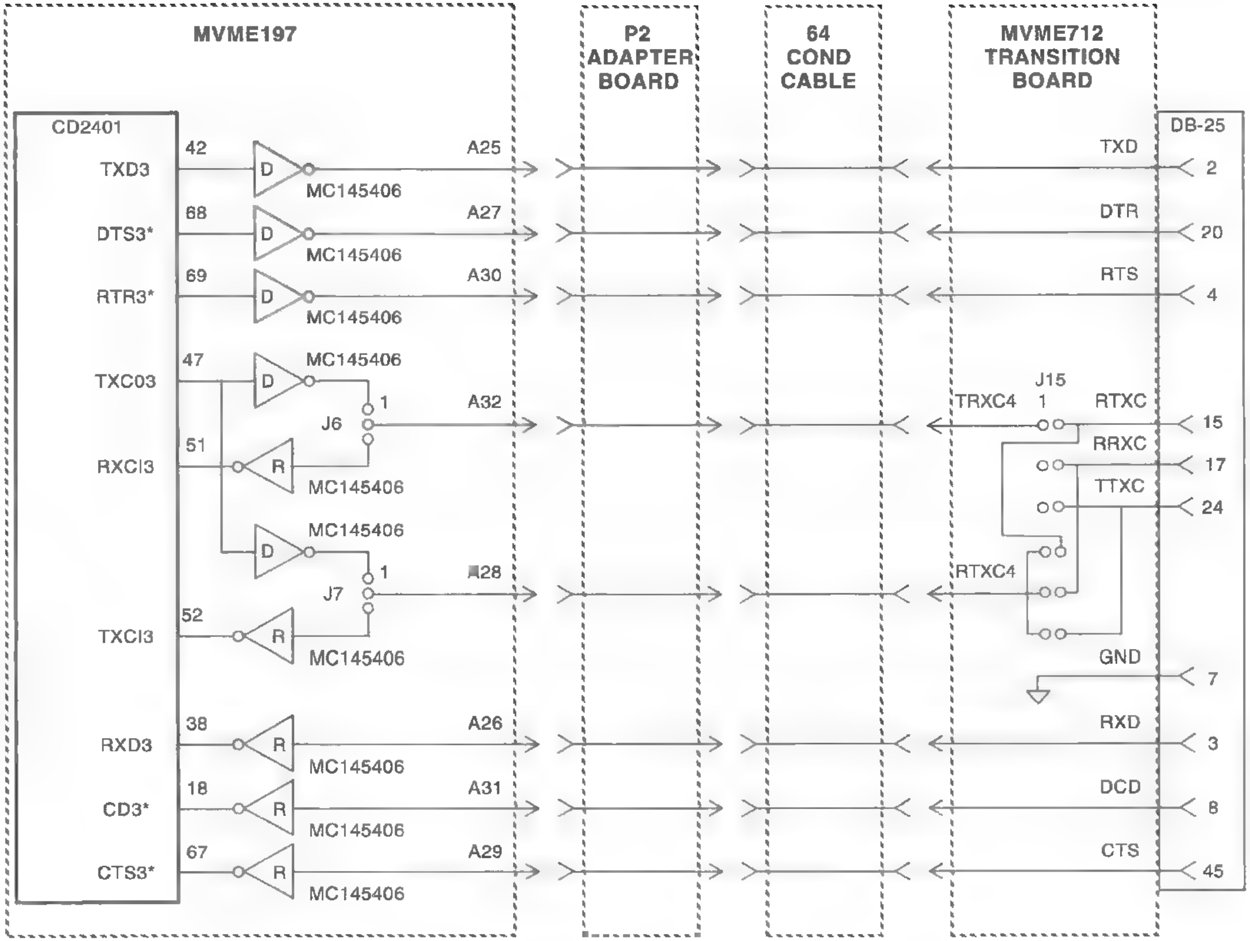


Figure 7-10. Serial Port 4 Configured as DTE

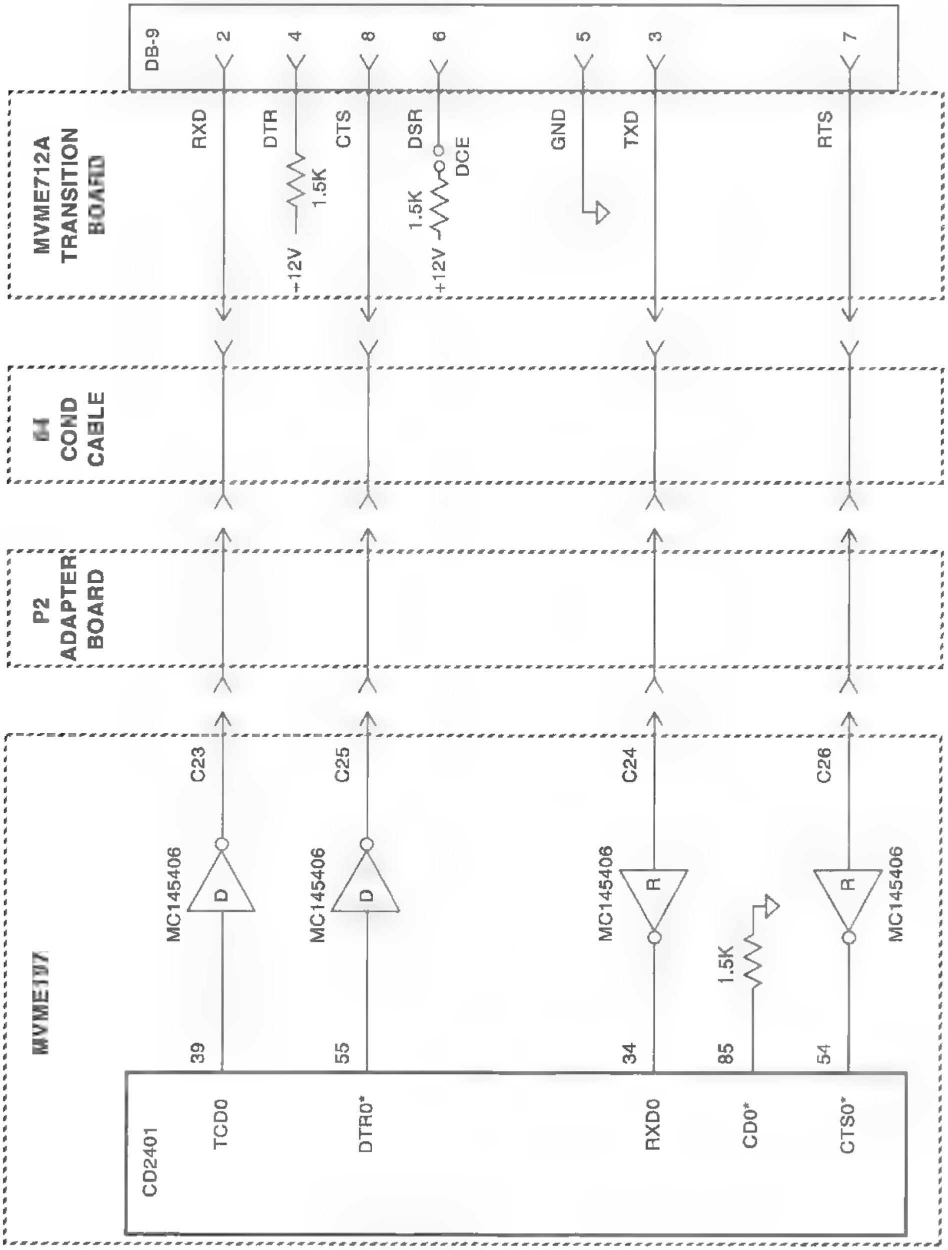


Figure 7-11. Serial Port 1 with MVME712A

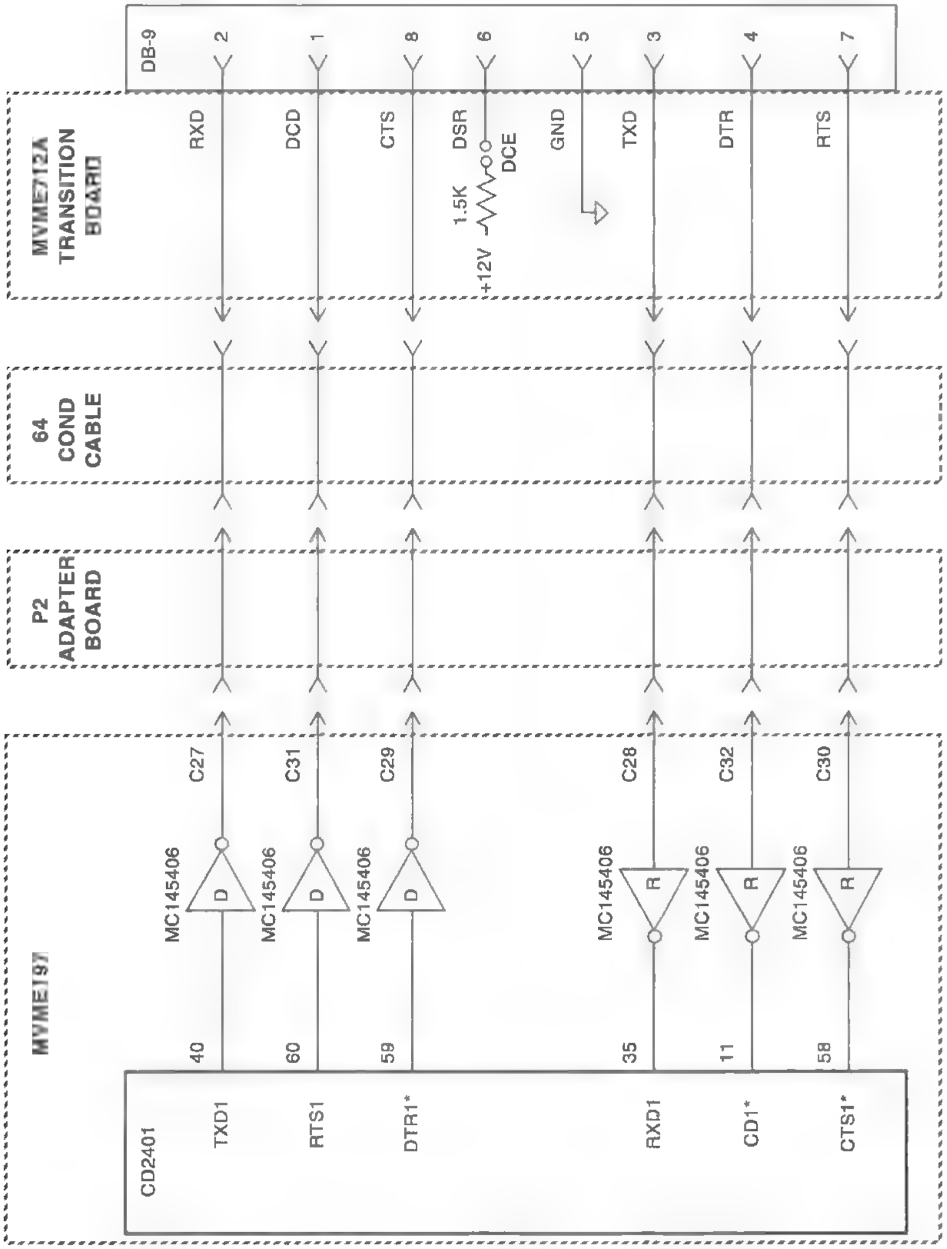


Figure 7-12. Serial Port 2 with MVME712A

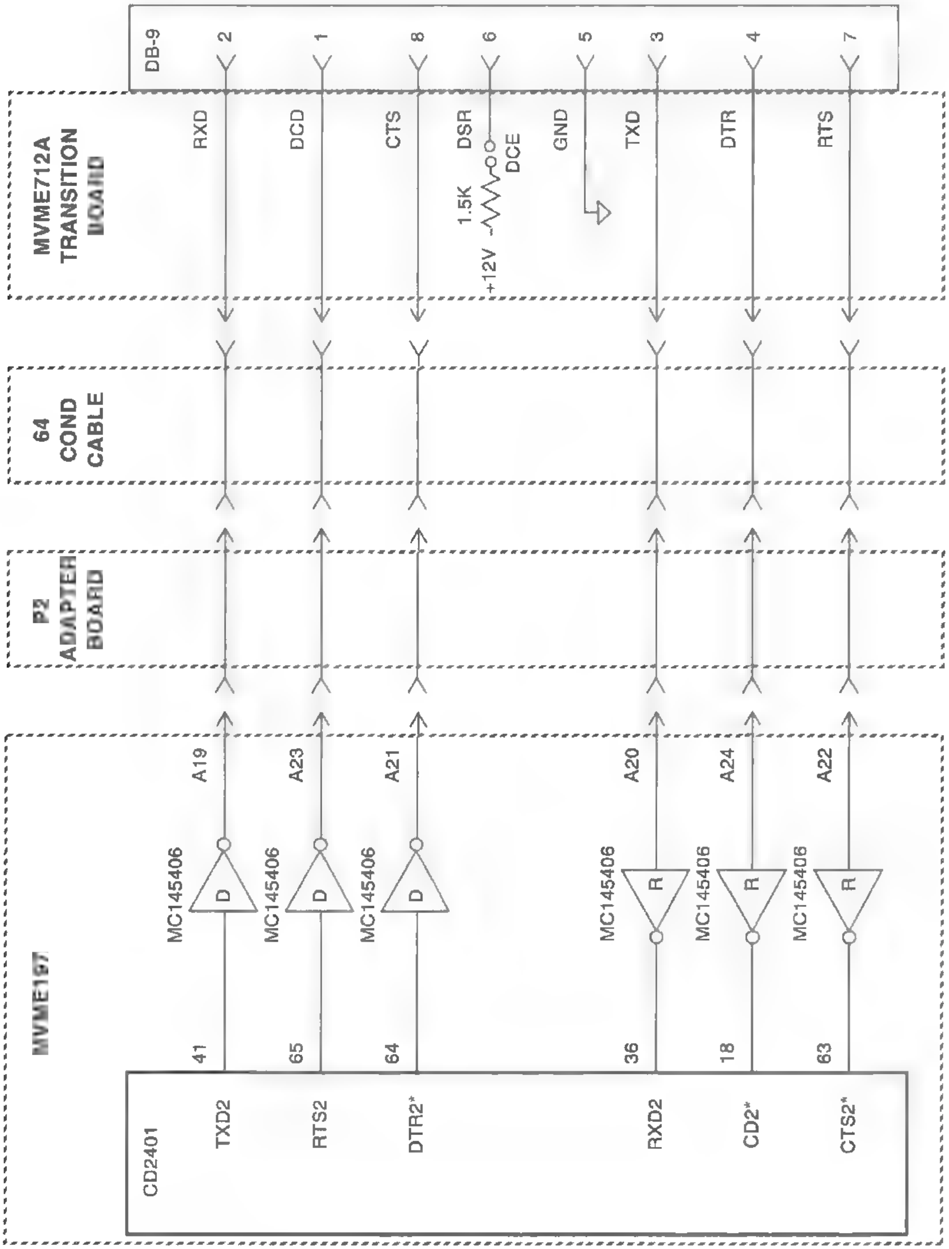


Figure 7-13. Serial Port 3 with MVME712A

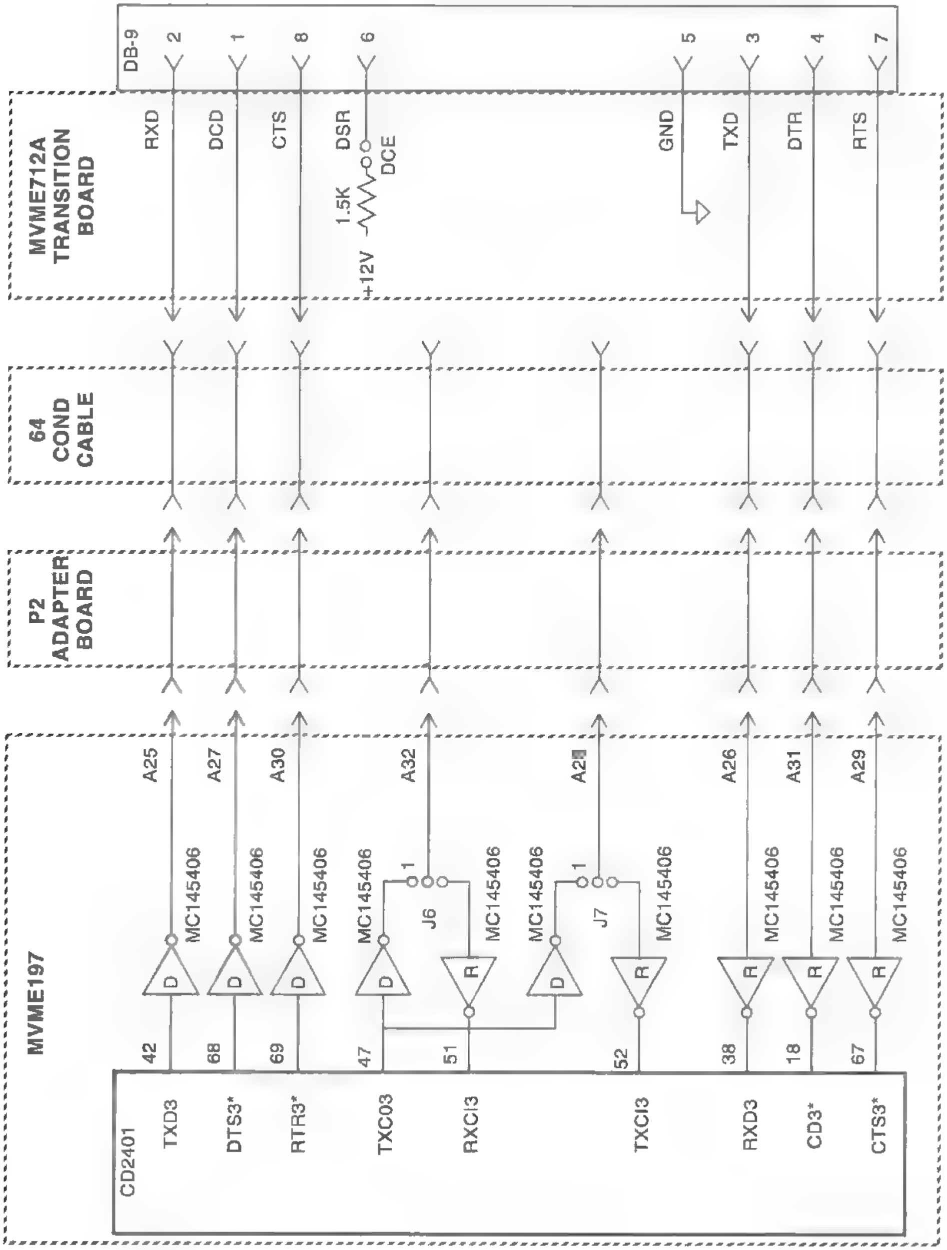


Figure 7-14. Serial Port 4 with MVME712A

Index

When using this index, keep in mind that a page number indicates only where referenced material begins. It may extend to the page or pages following the page referenced.

INDEX

INDEX

INDEX

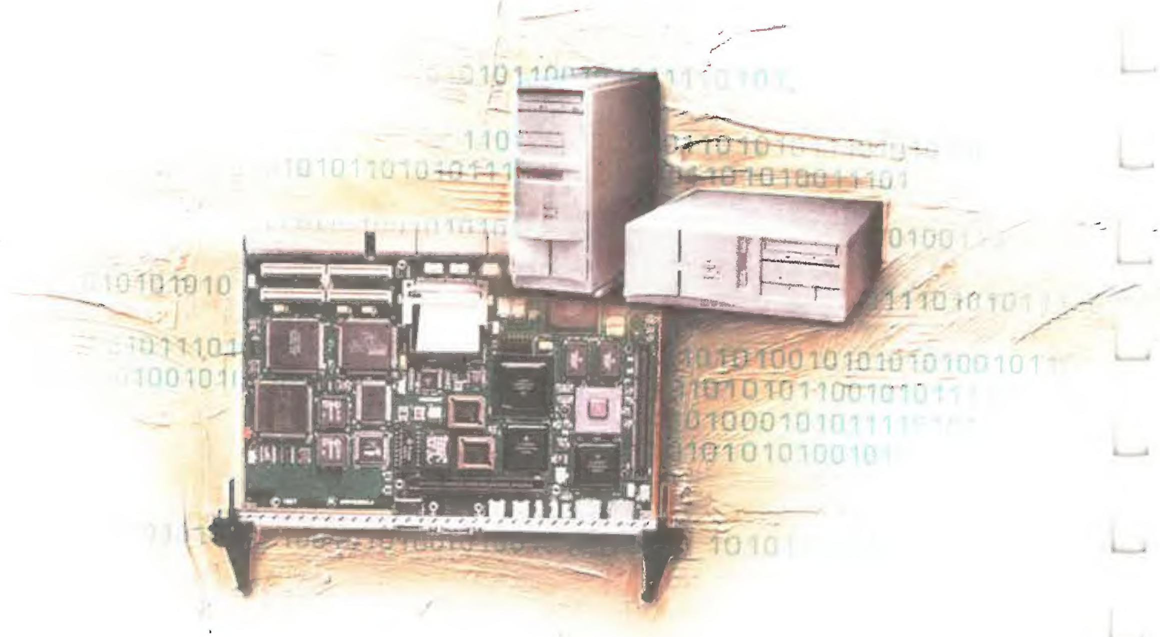
INDEX

INDEX

INDEX

INDEX

For electronic versions
and updates, visit
http://www.mcg.mot.com



MOTOROLA
Computer Group

2900 South Diablo Way
Tempe, Arizona 85282
P.O. Box 2953
Phoenix, Arizona 85062-2953